N° d'ordre : D.U.2343
EDSPIC : 606

UNIVERSITÉ BLAISE PASCAL - CLERMONT-FERRAND II

École Doctorale
Sciences Pour l'Ingénieur de Clermont-Ferrand

Thèse présentée par :
Mohamed TAMAAZOUSTI

Formation Doctorale CSTI :
Composants et Systèmes pour le Traitement de l'Information

en vue de l'obtention du grade de

DOCTEUR D'UNIVERSITÉ

Spécialité : Vision pour la robotique

L'ajustement de faisceaux contraint comme cadre d'unification des méthodes de localisation : application à la réalité augmentée sur des objets 3D

Soutenue publiquement le 13 mars 2013 devant le jury :

M. Michel DEVY	Président
M^me^ Marie-Odile BERGER	Rapporteur
M. Eric MARCHAND	Rapporteur
M. Vincent LEPETIT	Examinateur
M^me^ Sylvie NAUDET-COLLETTE	Examinateur
M. Vincent GAY-BELLILE	Examinateur
M. Michel DHOME	Directeur de thèse

Remerciements

Je souhaite commencer ce mémoire de thèse en remerciant vivement toutes les personnes qui ont rendu possible de près ou de loin l'aboutissement de mes travaux de thèse. Ce fut pour moi une aventure formidable aussi bien sur le plan humain que scientifique. Je souhaiterais donc tout d'abord remercier l'ensemble des personnes qui m'ont accueilli au sein de leur unité, notamment François Gaspard, aujourd'hui chef du département DIASI du CEA LIST.

Cette aventure de trois ans s'est clôturée comme à l'accoutumée par la rédaction d'un mémoire et par une soutenance. Pour cette dernière j'ai eu la chance d'avoir un jury exceptionnel dont je tiens à remercier solennellement et avec un profond respect et une grande considération les membres qui ont donné de leur temps précieux à l'évaluation de mes travaux. Je remercie, en particulier, Marie-Odile Berger et Eric Marchand pour avoir accepté de rapporter mes travaux et pour les échanges scientifiques que nous avons pu avoir. Je remercie également Michel Devy pour avoir présidé ma soutenance ainsi que Vincent Lepetit pour avoir examiné mes travaux. Les membres de mon jury ont par ailleurs su rendre cette journée particulièrement agréable pour moi.

De mon point de vue, lors de la réalisation d'une thèse, l'équipe d'encadrement influe indéniablement sur le déroulement de celle-ci ainsi que sur la façon dont cette expérience est vécue par le doctorant. Pour ma part, j'ai eu la chance d'être encadré par des personnes qui ont su me motiver au quotidien, et qui, forts de leurs expériences respectives et de leurs connaissances dans le domaine, ont joué un rôle important dans le déroulement de ma thèse. Je remercie donc mes encadrants du LVIC, Sylvie Naudet-Collette et Vincent Gay-Bellile, qui ont été présents pour moi au quotidien. Ma gratitude s'adresse aussi, bien entendu, à Michel Dhome mon directeur de thèse qui a su se rendre très disponible, aussi bien pour les nombreuses réunions d'avancement que pour la relecture du mémoire, et ce malgré un emploi du temps très chargé. Bien au-delà de l'aspect professionnel, se sont les qualités humaines de chacun que je voudrais souligner par ces quelques lignes.

Mes remerciements vont aussi vers mes anciens collègues de travail notamment les stagiaires, doctorants, postdocs, ingénieurs, chercheurs et secrétaires du laboratoire LVIC, pour leur professionnalisme, leur convivialité et leur bonne humeur communicative et pour avoir ainsi formé un environnement de travail efficace et agréable. Je remercie particulièrement les membres de l'équipe 3D avec qui j'ai pu

avoir de nombreux échanges scientifiques et techniques très enrichissants. Je voudrais notamment, remercier Steve Bourgeois pour son aide tout au long de ma thèse en particulier lors de la préparation de ma soutenance.

Par ailleurs, je souhaiterai remercier les personnes que je côtoie dans le cadre du projet Siforas sans oublier les membres de la société Diotasoft, en particulier Lionnel Joussemet avec qui j'ai eu la chance d'avoir des échanges très enrichissants.

J'aimerais maintenant adresser des remerciements plus personnels mais néanmoins essentiels. En effet, la thèse est une aventure professionnelle qui évolue en parallèle de la vie personnelle du doctorant et qui, de ce fait, a généralement un impact non négligeable sur cette dernière. Me concernant, le soutient que j'ai eu durant toute cette aventure de la part de ma famille a été indispensable. De manière plus générale, je remercie mes parents, Chaïb et Malika, pour avoir accepté tous mes choix, pour m'avoir toujours épaulé, mais aussi pour avoir mis à ma disposition l'ensemble des ressources affectives et matérielles nécessaires pour me permettre d'avancer dans la vie. Je tiens aussi à remercier mes frères et sœurs pour leur soutient durant mes études et leurs encouragements ainsi que pour m'avoir enseigné le sens des valeurs humaines et incité à toujours donner le meilleur de moi-même quelque soit ce que j'entreprends dans la vie.

Enfin, mes plus tendres remerciements vont vers Hajer, ma femme, pour le soutient et le réconfort qu'elle m'apporte, pour la patience et la compréhension dont elle fait toujours preuve et plus encore, pour le bonheur que je vis à ses côtés depuis notre mariage.

Résumé

Les travaux réalisés au cours de cette thèse s'inscrivent dans la problématique de localisation en temps réel d'une caméra par vision monoculaire. Dans la littérature, il existe différentes méthodes qui peuvent être classées en trois catégories. La première catégorie de méthodes considère une caméra évoluant dans un environnement complètement inconnu (SLAM). Cette méthode réalise une reconstruction en ligne de primitives observées dans des images d'une séquence vidéo et utilise cette reconstruction pour localiser la caméra. Les deux autres permettent une localisation par rapport à un objet 3D de la scène en s'appuyant sur la connaissance, *a priori*, d'un modèle de cet objet (suivi basé modèle). L'une utilise uniquement l'information du modèle 3D de l'objet pour localiser la caméra, l'autre peut être considérée comme l'intermédiaire entre le SLAM et le suivi basé modèle. Cette dernière méthode consiste à localiser une caméra par rapport à un objet en utilisant, d'une part, le modèle de ce dernier et d'autre part, une reconstruction en ligne des primitives de l'objet d'intérêt. Cette reconstruction peut être assimilée à une mise à jour du modèle initial (suivi basé modèle avec mise à jour). Chacune de ces méthodes possède des avantages et des inconvénients.

Dans le cadre de ces travaux de thèse, nous proposons une solution unifiant l'ensemble de ces méthodes de localisation dans un unique cadre désigné sous le terme de SLAM contraint. Cette solution, qui unifie ces différentes méthodes, permet de tirer profit de leurs avantages tout en limitant leurs inconvénients respectifs. En particulier, nous considérons que la caméra évolue dans un environnement partiellement connu, c'est-à-dire pour lequel un modèle (géométrique ou photométrique) 3D d'un objet statique de la scène est disponible. L'objectif est alors d'estimer de manière précise la pose de la caméra par rapport à cet objet 3D. L'information absolue issue du modèle 3D de l'objet d'intérêt est utilisée pour améliorer la localisation de type SLAM en incluant cette information additionnelle directement dans le processus d'ajustement de faisceaux. Afin de pouvoir gérer un large panel d'objets 3D et de scènes, plusieurs types de contraintes sont proposées dans ce mémoire. Ces différentes contraintes sont regroupées en deux approches. La première permet d'unifier les méthodes SLAM et de suivi basé modèle, en contraignant le déplacement de la caméra via la projection de primitives existantes extraites du modèle 3D dans les images. La seconde unifie les méthodes SLAM et de suivi basé modèle avec mise à jour en contraignant les primitives reconstruites par le SLAM à appartenir à la

surface du modèle (unification SLAM et mise à jour du modèle).

Les avantages de ces différents ajustements de faisceaux contraints, en terme de précision, de stabilité de recalage et de robustesse aux occultations, sont démontrés sur un grand nombre de données de synthèse et de données réelles. Des applications temps réel de réalité augmentée sont également présentées sur différents types d'objets 3D. Ces travaux ont fait l'objet de 4 publications internationales, de 2 publications nationales et d'un dépôt de brevet.

Mots clés : Vision par ordinateur, localisation et cartographie simultanées par vision, réalité augmentée, SLAM contraint, Ajustement de faisceaux contraint, suivi basé modèle.

Abstract

This thesis tackles the problem of real time location of a monocular camera. In the literature, there are different methods which can be classified into three categories. The first category considers a camera moving in a completely unknown environment (SLAM). This method performs an online reconstruction of the observed primitives in the images and uses this reconstruction to estimate the location of the camera. The two other categories of methods estimate the location of the camera with respect to a 3D object in the scene. The estimation is based on an *a priori* knowledge of a model of the object (Model-based). One of these two methods uses only the information of the 3D model of the object to locate the camera. The other method may be considered as an intermediary between the SLAM and Model-based approaches. It consists in locating the camera with respect to the object of interest by using, on one hand the 3D model of this object, and on the other hand an online reconstruction of the primitives of the latter. This last online reconstruction can be regarded as an update of the initial 3D model (Model-based with update). Each of these methods has advantages and disadvantages.

In the context of this thesis, we propose a solution in order to unify all these localization methods in a single framework referred to as the constrained SLAM, by taking parts of their benefits and limiting their disadvantages. We, particularly, consider that the camera moves in a partially known environment, *i.e.* for which a 3D model (geometric or photometric) of a static object in the scene is available. The objective is then to accurately estimate the pose (position and orientation) of the camera with respect to this object. The absolute information provided by the 3D model of the object is used to improve the localization of the SLAM by directly including this additional information in the bundle adjustment process. In order to manage a wide range of 3D objets and scenes, various types of constraints are proposed in this study and grouped into two approaches. The first one allows to unify the SLAM and Model-based methods by constraining the trajectory of the camera through the projection, in the images, of the 3D primitives extracted from the model. The second one unifies the SLAM and Model-based with update methods, by constraining the reconstructed 3D primitives of the object to belong to the surface of the model (unification SLAM and model update).

The benefits of the constrained bundle adjustment framework in terms of accuracy, stability, robustness to occlusions, are demonstrated on synthetic and real data.

Real time applications of augmented reality are also presented on different types of 3D objects. This work has been the subject of four international publications, two national publications and one patent.

Key-words : Computer vision, Simultaneous Localization and Mapping, Augmented Reality, Constrained SLAM, Constrained Bundle Adjustment, Model-based tracking.

Notations et acronymes

Mathématiques

E^n	Espace E de dimension n
M	Matrice
$\mathbf{v}$	Vecteur
$\mathbf{x} \cdot \mathbf{y}$	Produit scalaire entre les vecteurs $\mathbf{x}$ et $\mathbf{y}$
$\mathbf{x} \times \mathbf{y}$	Produit vectoriel entre les vecteurs $\mathbf{x}$ et $\mathbf{y}$
M^T	Transposée de la matrice M
$\mathcal{I}_n$	La matrice identité de dimension $(n \times n)$
$[\mathbf{x}]_\times$	Matrice antisymétrique créée à partir du vecteur $\mathbf{x}$, telle que $[\mathbf{x}]_\times = \begin{pmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{pmatrix}$ elle permet de réaliser le produit vectoriel $\mathbf{x} \times \mathbf{y}$ sous la forme du produit matriciel $[\mathbf{x}]_\times \mathbf{y} = \mathbf{x} \times \mathbf{y}$
M^+	Pseudo-inverse de la matrice M, telle que $\mathsf{M}^+ = (\mathsf{M}^\mathsf{T}\mathsf{M})^{-1}\mathsf{M}^\mathsf{T}$
f	Fonction mathématique

Géométrie euclidienne et projective

$\sim$	Egalité à un facteur non-nul près
$\mathbf{q}$	Point 2D
$\tilde{\mathbf{q}}$	Coordonnées homogènes du point 2D
$\mathbf{Q}$	Point 3D
$\tilde{\mathbf{Q}}$	Coordonnées homogènes du point 3D
d	Droite de l'espace
π	Plan de l'espace
R	Matrice de rotation
$\mathbf{t}$	Vecteur de translation
S	Transformation géométrique 3D
S	Matrice de transformation associée à S

Caméras et reconstruction 3D

$\mathcal{C}$	Caméra
I	Image capturée par la caméra
P	Matrice de projection
K	Matrice de calibrage
F	Matrice Fondamentale
E	Matrice Essentielle
C_j	j^e caméra reconstruite
$\mathbf{Q}_i$	i^e point 3D reconstruit
$\mathbf{q}_{ij}$	Observation du i^e point 3D par la j^e caméra

Acronymes

AVV	Asservissement Visuel Virtuel
CAO	Conception assistée par ordinateur
DoF	*Degree of Freedom* (degré de liberté)
EKF	*Extended Kalman Filter*
ICP	*Iterative Closest Point*
IRLS	*Iteratively Reweighted Least Squares*
MAD	*Median Absolute Deviation* (l'écart absolu à la médiane)
SfM	*Structure from Motion* (structure à partir du mouvement)
SLAM	*Simultaneous Localization And Mapping* (Localisation et cartographie simultanées)
UKF	*Unscented Kalman Filter*

Sommaire

Introduction

Cadre général

Dans l'industrie, les impératifs d'augmentation de la productivité et de la qualité ont entrainé une réduction des interventions humaines au profit de l'automatisation et de la robotisation des étapes de production et de contrôle qualité des produits manufacturés, par exemple, l'automatisation du contrôle d'aspect, la robotisation de l'assemblage, *etc.* Au cours de ces processus, l'étape de localisation de l'objet représente un élément crucial. Si une localisation 2D peut être suffisante pour des objets pouvant être considérés comme plan (par exemple une carte électronique), une localisation 3D (c'est à dire position 3D et orientation 3D) est, plus généralement, nécessaire pour les objets en volume.

Si cette automatisation permet de réduire certains coûts de production, l'intervention humaine reste indispensable pour de nombreuses étapes du cycle de vie d'un produit. Ce sont, par exemple, certaines étapes de production et de contrôle trop complexes, mais ce sont aussi, et surtout, les étapes de vente, d'exploitation et de maintenance du produit. Une réduction des coûts et un gain en performance associés à ces étapes peuvent être obtenus en proposant à l'opérateur des outils d'aide à la décision. En particulier, les systèmes de Réalité Augmentée (RA) semblent être des outils prometteurs puisqu'ils permettent d'augmenter en temps réel la vision que l'utilisateur a de l'objet avec toute information nécessaire à la prise de décision. Ainsi, la RA peut tout aussi bien, aider un vendeur à présenter l'ensemble des caractéristiques et options d'un produit pour faciliter sa vente, tout comme elle peut assister un opérateur de maintenance en lui indiquant les actions à effectuer étape par étape. Cependant, pour que l'information ajoutée par le système soit correctement positionnée par rapport à l'objet, une localisation 3D précise de ce dernier est essentielle. Or, si les systèmes de localisation exploités dans les étapes de production ou de contrôle fonctionnent généralement dans des environnements maîtrisés, avec des capteurs et des systèmes de calculs coûteux, il n'en est pas de même pour les étapes de maintenance, de vente et d'exploitation qui se déroulent généralement dans un environnement peu ou pas contrôlé et sur des plateformes mobiles. De plus cette localisation doit être réalisée en continu et en temps réel.

De nombreuses solutions de localisation sont exploitées dans le monde de l'industrie, celles-ci présentent cependant des limitations rédhibitoires à leur usage pour

les applications de réalité augmentée. Les systèmes utilisant des méthodes de localisation mécaniques (*eg.* la solution `FaroArm` de Faro) sont précis mais coûteux et limitent l'utilisateur à un volume restreint. De même, les méthodes de localisation magnétiques (*eg.* la solution `Patriot` de Polhemus), généralement moins coûteuses, sont vulnérables à la distorsion (perturbation) des signaux par les métaux de l'environnement et limitent aussi le domaine de déplacement. Les méthodes de localisation utilisant les ultrasons [1] divergent au cours du temps à cause des variations de la température ambiante et sont limitées par la fréquence de localisation. De plus, ces deux dernières méthodes nécessitent d'aménager l'environnement. Les méthodes de localisation utilisant un capteur inertiel dérivent avec le temps et proposent une localisation relative. Enfin, les solutions de localisation par vision par ordinateur peuvent être relativement précises, peu coûteuses, et permettent de gérer de grands volumes mais nécessitent souvent un environnement contrôlé (notamment cibles codées) pour garantir la précision et la robustesse de la mesure.

L'ensemble de ces méthodes présente donc des limitations critiques pour les systèmes de réalité augmentée pour un usage en conditions non maîtrisées (en point de vente, chez le client, *etc.*), ceci en raison des difficultés de mise en œuvre (l'aménagement de l'environnement, calibrage, *etc.*) ainsi que du coût du système. Cela limite l'offre du service d'assistance par RA au niveau de la vente, de l'exploitation et de la maintenance. Cependant la vision par ordinateur semble avoir le plus grand potentiel pour fournir une solution de RA, d'autant plus que les terminaux mobiles actuels (les tablettes voire les *smartphones*) offrent une plateforme adaptée à ce type de solutions car ils disposent d'une caméra et de la puissance de calcul nécessaire.

Problématiques et objectifs de la thèse

Si de nombreuses solutions de localisation 3D par vision monoculaire pour la réalité augmentée existent, aucune solution générique ne permet de répondre à l'ensemble des exigences requises par l'industrie :

- ▷ précision : la localisation doit permettre un alignement (ou recalage) précis et stable (sans effet de *jittering*) des éléments virtuels sur l'objet dans les images ;
- ▷ robustesse : la précision de localisation ne doit être altérée ni par les changements d'aspect de l'objet liés à son usure, ni par les variations des conditions d'éclairage, ou de point de vue ;
- ▷ fréquence de localisation : le flux vidéo doit être traité en temps réel ;
- ▷ un environnement peu ou pas contrôlé : peu d'hypothèses peuvent être faites sur la nature de l'environnement ou des conditions d'éclairage. D'autre part, il ne peut être exclu que des occultations surviennent au cours de l'utilisation du système ;

1. `http://isd.ktu.lt/it2011//material/Proceedings/5_ITA_4.pdf`

- ▷ souplesse : la méthode de localisation doit permettre de localiser des objets de différentes natures (texturés ou non, polyédriques ou non, *etc.*) afin de couvrir le plus de cas d'usage possible, tout en évitant le coût lié à une multiplication des solutions de localisation à intégrer. Les solutions actuelles sont généralement limitées à des objets d'une nature précise ;
- ▷ facilité d'utilisation : la solution doit pouvoir être utilisée par des non spécialistes et ne doit donc pas nécessiter l'intervention de l'utilisateur pour aménager l'environnement ou encore pour réaliser des processus complexes de calibrage ou de reconstruction ;
- ▷ facilité de déploiement : la solution doit pouvoir être déployée sur des plateformes mobiles telles que les tablettes, voir les *smartphones*, afin de couvrir un large marché ;
- ▷ données nécessaires : si les solutions de localisation reposent généralement sur l'exploitation d'un modèle 3D de l'objet (*eg.* un modèle de conception assistée par ordinateur CAO), ces modèles ne sont pas toujours accessibles. La diffusion des modèles CAO est généralement limitée pour éviter la reproduction frauduleuse des objets industriels. L'algorithme doit donc être capable de fonctionner avec des modèles précis mais aussi des modèles reconstruits de manière plus approximative tels que des modèles reconstruits par une Kinect (capteur actif destiné à la console de jeux vidéo `Xbox` de Microsoft).

L'objectif principal de cette thèse est de proposer un cadre unique de localisation 3D par vision monoculaire répondant à l'ensemble de ces exigences. Pour cela nous proposons une solution utilisant une seule caméra ainsi qu'un modèle 3D d'un objet d'intérêt de la scène.

Contributions de la thèse

Dans cette thèse, nous proposons un cadre permettant d'unifier les méthodes de suivi d'objet connu (suivi basé modèle) avec les méthodes d'estimation de mouvement d'une caméra en environnement inconnu (SLAM monoculaire). En effet, alors que les méthodes de localisation actuelles exploitent généralement soit uniquement l'information visuelle apportée par l'objet d'intérêt dont un modèle 3D est disponible, soit l'information visuelle de l'ensemble de la scène observée lorsqu'aucune connaissance *a priori* n'est disponible, nous considérons le cas d'une localisation en environnement partiellement connu, c'est à dire un environnement inconnu dans lequel se trouve un objet d'intérêt dont le modèle 3D est connu. Nous faisons alors l'hypothèse que l'objet est statique dans la scène. Pour que le processus de localisation bénéficie à la fois de l'information apportée par l'objet connu et de l'environnement inconnu, nous proposons de reprendre le cadre des méthodes de localisation et cartographie simultanées par images clés, plus généralement connues sous le nom de *Keyframe-based Simultaneous Localisation and Mapping* (ou *Keyframe* SLAM), et d'y introduire des contraintes supplémentaires issues de la connaissance

a priori de l'objet. La solution ainsi obtenue correspond à un **SLAM contraint**. Plus précisément, nous proposons d'introduire ces contraintes au niveau de l'ajustement de faisceaux afin d'assurer une localisation précise de la caméra ainsi qu'une reconstruction précise de l'environnement. Dans le but de montrer que ce cadre d'**ajustement de faisceaux contraint** est générique, différentes contraintes sont proposées permettant ainsi de traiter des objets de différentes natures.

Outre les contributions scientifiques présentées ci-dessus, cette thèse contient également des contributions techniques relatives à l'intégration de notre méthode de localisation dans un système de RA pour l'industrie : les problèmes d'initialisation et de création d'un modèle 3D de l'objet sont abordés. De nombreuses expérimentations sur des objets de natures variées (voiture, cuisine, pièce industrielle, *etc.*) sont aussi présentées.

Les travaux réalisés durant cette thèse ont donné lieu à différentes contributions scientifiques et techniques qui ont été valorisées par quatre publications internationales (Tamaazousti *et al.* (2011b,c); Besbes *et al.* (2012); Gay-Bellile *et al.* (2012)), deux publications nationales (Tamaazousti *et al.* (2011d, 2012)) et un dépôt de brevet (Tamaazousti *et al.* (2011a)). De plus ces travaux font actuellement au sein du laboratoire, l'objet d'un transfert industriel.

Contexte de la thèse

Cette thèse a été effectuée entre novembre 2009 et novembre 2012 au Laboratoire Vision et Ingénierie des Contenus (LVIC) du CEA LIST, à Saclay. L'ensemble des travaux ont été réalisés en cotutelle avec le laboratoire Institut Pascal (Unité Mixte de Recherche - UBP - CNRS - IFMA), à Clermont-Ferrand.

Organisation du mémoire

En premier lieu, le chapitre 1 présente l'ensemble des notations et des outils de base nécessaires à la bonne compréhension du mémoire. Le chapitre 2 est un état de l'art sur les méthodes de localisation par vision monoculaire. Il présente et compare les méthodes de type SLAM « classique » et les méthodes basées modèle. La structure de la suite du mémoire est guidée par la solution générale proposée pour la localisation en temps réel d'une caméra par rapport à un objet d'intérêt. Le principe ainsi que le formalisme proposé pour cette approche sont présentés dans le chapitre 3. Il s'agit d'une méthode de SLAM dans laquelle l'ajustement de faisceaux est contraint par des informations liées à la connaissance *a priori* d'un modèle 3D. Les chapitres 4 et 5 déclinent deux formulations possibles de l'approche proposée. La première est une méthode permettant d'unifier le SLAM avec les méthodes de localisation par rapport à un modèle 3D. La seconde unifie le SLAM avec les méthodes de localisation avec mise à jour du modèle. Le chapitre 6 présente des

applications de réalité augmentée s'appuyant sur la méthode de localisation proposée. Nous dressons enfin un bilan des travaux réalisés et présentons différentes perspectives envisageables pour les travaux futurs.

CHAPITRE

1 Notions de base

Dans ce chapitre, nous présentons les notions de base et notations nécessaires à la compréhension du mémoire. Dans un premier temps nous introduisons les outils de vision par ordinateur concernant la représentation et la paramétrisation d'une caméra, nous mentionnons également quelques méthodes permettant d'estimer la géométrie d'une scène. Dans un second temps nous présentons quelques outils mathématiques utiles pour la compréhension du reste du manuscrit. Ce chapitre n'est pas destiné à une lecture exhaustive mais il représente un complément au mémoire. Plus de détails sur ces notions peuvent être trouvés dans le livre de Hartley et Zisserman (2000).

1.1 Caméra perspective et géométrie associée

1.1.1 Géométrie projective

On définit l'espace projectif de dimension n, noté $\mathbb{P}^n$, comme l'ensemble des classes d'équivalence de $\mathbb{R}^{n+1}$ formé pour la relation $\sim$ telle que :

$$\mathbf{u}, \mathbf{v} \in \mathbb{R}^{n+1*}, \mathbf{u} \sim \mathbf{v} \iff \exists \lambda \in \mathbb{R}^* / \mathbf{u} = \lambda \mathbf{v}. \tag{1.1}$$

Des études théoriques de ces espaces existent, nous nous intéresserons dans nos travaux uniquement à la géométrie projective qui leur est associée et qui permet en particulier de formaliser la notion de point à l'infini dans les espaces affines.

Un vecteur de l'espace projectif $\mathbb{P}^n$ aura pour coordonnées :

$$\tilde{\mathbf{q}} = [q_1 \ \dots \ q_{n+1}]^\mathsf{T}, \tag{1.2}$$

avec les q_i non tous nuls. Si le dernier paramètre q_{n+1} est non nul, alors $\mathbf{q}$ représente le vecteur $\tilde{\mathbf{q}}$ de $\mathbb{R}^n$ avec $\mathbf{q} = [q_1/q_{n+1} \ \dots \ q_n/q_{n+1}]^\mathsf{T}$. Dans le cas contraire, le vecteur $\tilde{\mathbf{q}}$ décrit un point à l'infini. Les coordonnées $\tilde{\mathbf{q}}$ sont appelées coordonnées homogènes de $\mathbf{q}$. Dans l'ensemble du mémoire, l'utilisation du tilde indiquera que

les coordonnées utilisées sont homogènes. La fonction qui permet de passer des coordonnées homogènes aux coordonnées euclidiennes est notée Ψ et est définie par :

$$\begin{array}{rccc} \Psi & : & \mathbb{P}^n & \to & \mathbb{R}^n \\ & & [q_1 \ \dots \ q_{n+1}]^\mathsf{T} & \mapsto & [q_1/q_{n+1} \ \dots \ q_n/q_{n+1}]^\mathsf{T}. \end{array} \tag{1.3}$$

1.1.2 Représentation d'une caméra

Nous nous intéressons ici uniquement au modèle de caméra *pinhole* qui est adapté à la plupart des caméras actuellement utilisées en vision par ordinateur. A noter que d'autres caméras deviennent de plus en plus populaires notamment les caméras omnidirectionnelles dont la représentation se prête à un traitement mathématique similaire à celui présenté ci-dessous. Nous présentons alors uniquement les méthodes permettant de comprendre la suite du document. Pour plus de détails sur les modèles de caméra et sur l'étude plus approfondie des propriétés numériques et géométriques du calibrage de caméra, nous renvoyons de nouveau le lecteur intéressé au livre de Hartley et Zisserman (2000).

1.1.2.1 Le modèle de projection perspective

Mathématiquement, la formation d'image peut être définie comme la projection d'un élément de l'espace 3D dans le plan image, comme illustré sur la figure 1.1. Les coordonnées d'un point 3D $\mathbf{Q} = [X, Y, Z]^\mathsf{T}$ exprimées en coordonnées euclidiennes et son point 2D $\mathbf{q} = [u, v]^\mathsf{T}$ correspondant dans l'image sont alors liées par l'équation suivante :

$$s\tilde{\mathbf{q}} = \mathsf{P}\tilde{\mathbf{Q}}, \tag{1.4}$$

où s est un facteur d'échelle, $\tilde{\mathbf{q}} = [u, v, 1]^\mathsf{T}$ et $\tilde{\mathbf{Q}} = [X, Y, Z, 1]^\mathsf{T}$ sont les coordonnées homogènes des points $\mathbf{q}$ et $\mathbf{Q}$, et P est une matrice de projection 3×4 définit à un facteur d'échelle près, et qui dépend donc de 11 paramètres. Elle est souvent considérée comme étant la matrice de projection perspective car elle décrit de manière réaliste et simple le comportement d'une caméra de qualité raisonnable. Une telle matrice de projection perspective peut être décomposée comme suit :

$$\mathsf{P} = \mathsf{K} \ (\mathsf{R} \mid \mathbf{t}) \,, \tag{1.5}$$

où :

- ▷ K est la matrice de calibrage 3×3 qui dépend des paramètres intrinsèques de la caméra telle que la longueur focale ;
- ▷ $(\mathsf{R} \mid \mathbf{t})$ est la matrice 3×4 des paramètres extrinsèques, et correspond à la transformation euclidienne permettant de passer du repère monde au repère de la caméra : R représente une matrice de rotation 3×3, et $\mathbf{t}$ un vecteur de translation 3×1 (voir section 1.1.3 pour plus de détails).

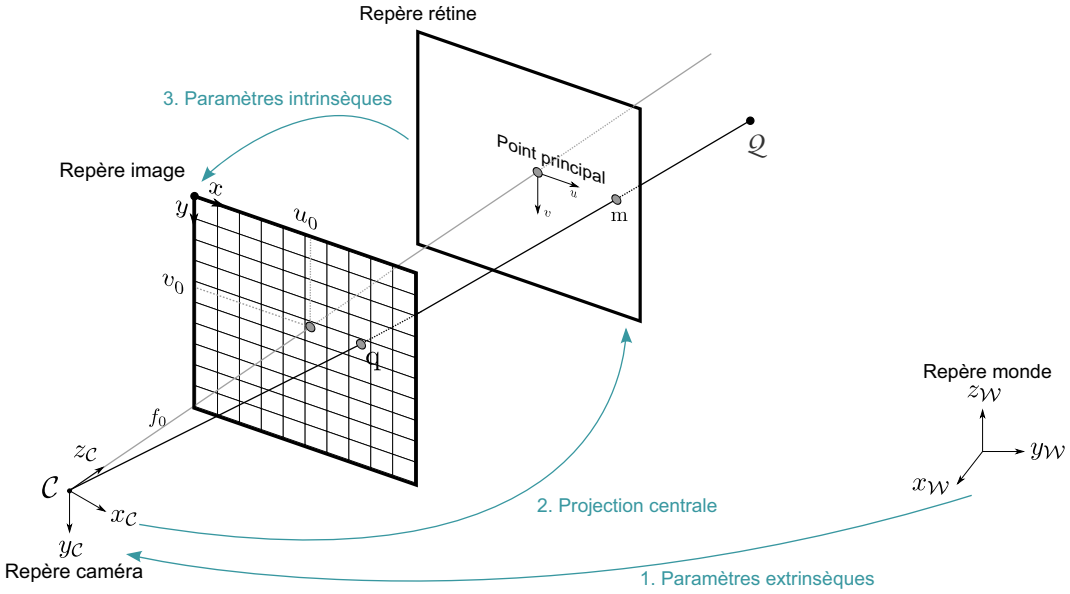


FIGURE 1.1 – **Projection perspective.** La projection perspective peut être vue comme trois transformations géométriques consécutives pour les points 3D.

1.1.2.2 La matrice de calibrage de la caméra

La matrice de calibrage K contient les paramètres intrinsèques de la caméra, également appelés paramètres internes. Nous notons

$$\mathsf{K} = \begin{pmatrix} \alpha_u & s & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{1.6}$$

où :

- ▷ α_u et α_v sont respectivement les facteurs d'échelle dans les directions u et v. Ils sont proportionnels à la longueur focale f de la caméra : $\alpha_u = k_u f$ et $\alpha_v = k_v f$, où k_u et k_v correspondent au nombre de pixels par unité de distance dans les directions u et v ;
- ▷ $\mathbf{c} = [u_0, v_0]^\mathsf{T}$ représente les coordonnées image de l'intersection de l'axe optique et du plan image, également appelé point principal ;
- ▷ s, est un facteur d'obliquité. Il est non nul si et seulement si les directions u et v sont non perpendiculaires, ce qui est extrêmement rare pour la plupart des caméras.

Le fait de considérer que le point principal $\mathbf{c}$ se trouve au centre de l'image est souvent une approximation raisonnable. De manière similaire si les pixels sont considérés comme étant carrés, α_u et α_v peuvent être considérés égaux. Dans la suite nous appellerons caméra calibrée toute caméra dont les paramètres intrinsèques sont connus.

Estimation de la matrice de calibrage de la caméra. Dans la plupart des méthodes de localisation par vision, les paramètres internes de la caméra sont considérés fixes et connus. Cela signifie qu'on ne peut pas changer le zoom de la caméra car il devient alors difficile de distinguer un changement au niveau de la longueur focale d'une translation le long de l'axe Z de la caméra. Ces paramètres peuvent être estimés à partir d'images acquises par la caméra lors d'une étape de calibrage réalisée hors ligne. Une discussion plus approfondie concernant les techniques de calibrage de caméra serait inutile dans ce mémoire. Nous nous contentons ici de préciser que les méthodes classiques de calibrage utilisent une mire de taille connue qui est positionnée dans le champ de vue de la caméra. Parfois, il s'agit d'une grille de calibrage 3D sur laquelle des motifs sont dessinés tels que ceux représentés dans la figure 1.2. Dans cet exemple, les coordonnées 3D des points blancs par rapport au coin au milieu de la grille sont connues de manière exacte. Il est alors facile de retrouver ces points dans l'image, et à partir des correspondances entre les points 3D et les points 2D image, de calculer les paramètres de la matrice de projection. Sturm et Maybank (1999) ont introduit une méthode de calibrage qui utilise une grille planaire observée de différents points de vue. Ces méthodes sont très flexibles car les motifs peuvent être simplement imprimés, puis collés à un objet planaire qui sera ensuite positionné en face de la caméra.

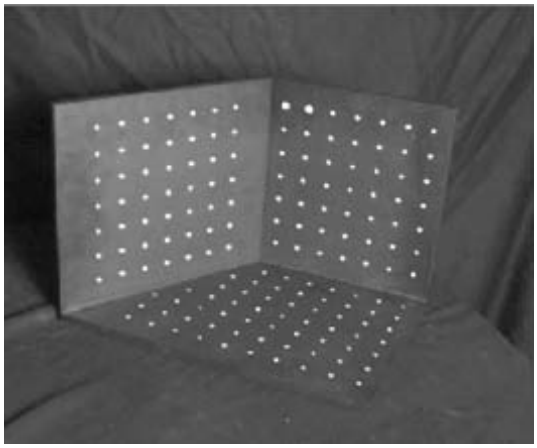


FIGURE 1.2 – Une grille 3D de calibrage utilisée pour l'estimation de la matrice de calibrage de la caméra.

Gestion de la distorsion de l'optique. Le modèle de projection perspective n'est pas toujours suffisant pour représenter tous les aspects de la formation de l'image car il ne tient pas compte de la distorsion possible de l'optique de la caméra, qui peut être non négligeable, tout particulièrement pour les caméras grand angle. La distorsion de l'optique peut être modélisée comme une déformation 2D de l'image. Etant donné une estimation des paramètres de distorsion, l'effet de distorsion peut

être éliminé (figure 1.3) en temps réel de manière efficace à chaque image capturée par la caméra en utilisant une table de correspondances. Une fois la distorsion corrigée, le modèle de projection perspective peut ensuite être utilisé.

(a) (b)

FIGURE 1.3 – Correction de la distorsion d'une image. (a) En raison de la distorsion, des projections de droites apparaissent incurvées dans l'image. (b) Dans l'image les distorsions étant corrigées, les projections des droites apparaissent maintenant droites.

Une représentation courante des paramètres de distorsion est comme suit : soit $\breve{\mathbf{u}} = [\breve{u}, \breve{v}]^\mathsf{T}$ les coordonnées distordues d'un pixel dans l'image et $\breve{\mathbf{x}} = [\breve{x}, \breve{y}]^\mathsf{T}$ les coordonnées normalisées correspondantes de sorte que $\breve{u} = u_0 + \alpha_u \breve{x}$ et $\breve{v} = v_0 + \alpha_v \breve{y}$ où u_0, v_0, α_u, et α_v sont les paramètres intrinsèques de l'équation (1.6). Soient respectivement $\mathbf{u} = (u, v)$ et $\mathbf{x} = (x, y)$ les valeurs corrigées en distorsion correspondantes à $\breve{\mathbf{u}}$ et $\breve{\mathbf{x}}$. La distorsion est exprimée comme la somme des éléments suivants : $\breve{\mathbf{x}} = \mathbf{x} + \mathbf{dx}_{radiale} + \mathbf{dx}_{tangentielle}$. La distorsion radiale peut être approximée comme suit :

$$\mathbf{dx}_{radiale} = (1 + k_1 r^2 + k_2 r^4 + ...)\mathbf{x}, \tag{1.7}$$

où $r = \|x\| = \sqrt{x^2 + y^2}$. La distorsion tangentielle $\mathbf{dx}_{tangentielle}$ a beaucoup moins d'influence, et peut être exprimée comme suit :

$$\mathbf{dx}_{tangentielle} = \begin{pmatrix} 2p_1 xy + p_2(r^2 + 2x^2) \\ p_1(r^2 + 2y^2) + 2p_2 xy \end{pmatrix}, \tag{1.8}$$

mais elle est généralement ignorée. Le logiciel « *Open Source Computer Vision Library* » permet d'estimer les paramètres de distorsion en utilisant une méthode dérivée de Heikkila et Silven (1997). C'est une méthode pratique pour des scènes de petites tailles.

1.1.3 Paramétrisation d'une pose 3D

1.1.3.1 Matrice des paramètres extrinsèques

La matrice des paramètres extrinsèques $(\mathsf{R} \mid \mathbf{t})$ de taille 3×4 représente l'orientation et la position de la caméra. Elle est formée de la matrice de rotation R et du vecteur de translation $\mathbf{t}$ et nous y ferons souvent référence en parlant de pose de caméra. Les applications de localisation par vision considèrent souvent que la matrice de calibrage K est connue et estiment uniquement R et $\mathbf{t}$. De manière plus formelle, cela correspond à la transformation euclidienne permettant de passer du repère monde au repère de la caméra. En effet, un point 3D représenté par le vecteur $\mathbf{Q}_{\mathcal{C}}$ dans le repère caméra s'écrit $\mathbf{Q}_{\mathcal{W}} = \mathsf{R}\mathbf{Q}_{\mathcal{C}} + \mathbf{t}$ dans le repère monde et à l'inverse, un point représenté par le vecteur $\mathbf{Q}_{\mathcal{W}}$ dans le repère monde s'écrit $\mathbf{Q}_{\mathcal{C}} = \mathsf{R}^{\mathsf{T}} (\mathbf{Q}_{\mathcal{W}} - \mathbf{t})$ dans le repère caméra.

Dans un but d'estimation ou d'optimisation numérique, la pose de la caméra doit être paramétrisée de manière appropriée. Alors qu'il est facile de représenter la translation de la caméra, paramétriser la rotation dans $\mathbb{R}^3$ est plus difficile. Une rotation dans $\mathbb{R}^3$ a uniquement trois degrés de liberté, et il est peu commode d'utiliser directement les neuf éléments de la matrice de rotation 3×3 qui représentent ses paramètres. De plus, pour s'assurer que la matrice est orthonormale, six contraintes doivent être ajoutées : trois pour forcer les trois colonnes à être unitaires, et trois autres pour les conserver mutuellement orthogonales.

Dans la suite, nous allons faire un bref rappel des différentes paramétrisations habituellement utilisées pour la localisation d'une caméra : les angles d'Euler, les quaternions et la paramétrisation angle/axe utilisant l'application exponentielle (*exponential map* en anglais). Pour ces trois paramétrisations, il existe des singularités, qui peuvent être évitées en reparamétrant localement la rotation. Néanmoins, nous verrons qu'en général, la dernière a les meilleures propriétés pour notre cadre d'étude concernant la localisation. Toutefois, si l'on se limite à de petites rotations, elles sont toutes équivalentes, car elles produisent la même approximation au premier ordre. Une comparaison plus approfondie peut être trouvée dans Grassia (1998).

1.1.3.2 Paramétrisation d'une rotation 3D

Définition. Les matrices orthonormales sont soumises aux contraintes d'orthonormalité :

$$\mathsf{R} \in O(p) \Leftrightarrow \mathsf{R}^{\mathsf{T}}\mathsf{R} = \mathcal{I}_{(p\times p)}. \tag{1.9}$$

Elles ont un déterminant dont la valeur absolue est 1. Les matrices orthonormales préservent, entre autres, la norme L_2 :

$$\mathsf{R} \in O(p) \Leftrightarrow \forall \mathbf{x} \in \mathbb{R}^p,\ \|\mathbf{x}\| = \|\mathsf{R}\mathbf{x}\|\,. \tag{1.10}$$

Les matrices de rotation forment un sous-groupe dans lequel les matrices ont un déterminant positif, c'est à dire dont la valeur est 1.

$$\mathsf{R} \in SO(p) \Leftrightarrow \mathsf{R}^{\mathsf{T}}\mathsf{R} = \mathcal{I}_{(p\times p)},\ \det(\mathsf{R}) = 1. \tag{1.11}$$

Angles d'Euler. Une matrice de rotation R peut toujours s'écrire comme le produit de trois matrices représentant les rotations autour des axes x, y et z. Il existe plusieurs conventions sur l'ordre dans lequel ces rotations doivent être effectuées. Par exemple, si on note α, β, γ comme étant respectivement les angles de rotations autour des axes z, y et x on a alors

$$\mathsf{R} = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{pmatrix}. \tag{1.12}$$

L'opération inverse, consistant en l'extraction des angles d'Euler pour une matrice de rotation donnée, peut être effectuée en identifiant les coefficients de la matrice à partir de leur expression analytique.

Même si les angles d'Euler peuvent constituer une bonne paramétrisation pour une large gamme d'applications, ils présentent néanmoins un inconvénient bien connu : lorsque deux des trois axes de rotations sont alignés, une des rotations n'a pas d'effet. En effet, comme les rotations sont appliquées successivement, il est alors possible que deux axes de rotations soient alignés. Ce problème est connu sous le nom de blocage de cardan (*gimbal lock* en anglais), et cette singularité peut mener à des problèmes d'optimisation mal conditionnés. Les représentations décrites ci-dessous sont conçues pour éviter ce problème.

Quaternions. Une rotation dans l'espace 3D peut également être représentée par un quaternion unitaire (Gennery (1992)). Les quaternions sont des nombres hypercomplexes qui peuvent être écrits comme une combinaison linéaire $a+bi+cj+dk$, avec $i^2 = j^2 = k^2 = ijk = -1$. Ils peuvent également être interprétés comme un couple $(a, \mathbf{v})$ formé d'un scalaire et d'un vecteur 3×1. Une rotation autour du vecteur unitaire $\mathbf{w}$ d'un angle θ est représentée par un quaternion unitaire

$$q = \left(\cos\left(\frac{1}{2}\theta\right), \mathbf{w}\sin\left(\frac{1}{2}\theta\right)\right). \tag{1.13}$$

Pour appliquer cette rotation à un point 3D $\mathbf{Q}$, il faut l'écrire sous la forme d'un quaternion $p = (0, \mathbf{Q})$. Le point après la rotation devient p' défini de la manière suivante :

$$p' = q \cdot p \cdot \overline{q}, \tag{1.14}$$

où $\cdot$ est l'opérateur multiplicatif des quaternion et $\overline{q} = \left(\cos\left(\frac{1}{2}\theta\right), -\mathbf{w}\sin\left(\frac{1}{2}\theta\right)\right)$ est le conjugué de q.

Cette représentation évite le problème de blocage de cardan, mais les techniques d'estimation doivent contraindre la norme de q afin qu'elle reste égale à un. Cela tend à augmenter la complexité algorithmique et n'est donc pas souhaitable en général. Nous décrivons par la suite la représentation axe/angle qui évite à la fois le problème de blocage de cardan et l'ajout de contraintes supplémentaires.

Axe/angle (*Exponential Map*). Contrairement à la représentation par les quaternions décrite précédemment, celle-ci ne nécessite que trois paramètres pour décrire une rotation. Elle ne souffre pas du problème de blocage de cardan et ses singularités se produisent dans une région de l'espace des paramètres qui peut facilement être évitée.

Soit $\mathbf{w} = [w_x, w_y, w_z]^\mathsf{T}$ un vecteur 3D et $\theta = \|\mathbf{w}\|$ sa norme. Une rotation angulaire θ autour d'un axe de direction $\mathbf{w}$ peut être représentée par une série infinie

$$\exp([\mathbf{w}]_\times) = \mathcal{I}_3 + [\mathbf{w}]_\times + \frac{1}{2!}[\mathbf{w}]_\times^2 + \frac{1}{3!}[\mathbf{w}]_\times^3 + ... \tag{1.15}$$

où $[\mathbf{w}]_\times$ est la matrice antisymétrique suivante

$$[\mathbf{w}]_\times = \begin{pmatrix} 0 & -w_z & w_y \\ w_z & 0 & -w_x \\ -w_y & w_x & 0 \end{pmatrix}. \tag{1.16}$$

Il s'agit de la représentation de *l'exponential map*, qui doit son nom au fait que la série de l'équation (1.15) est de la même forme que le développement en série d'une exponentielle. Elle peut être obtenue en utilisant la formule de Rodrigues,

$$\mathsf{R}([\mathbf{w}]_\times) = \exp([\mathbf{w}]_\times) = \mathcal{I}_3 + \sin\theta[\hat{\mathbf{w}}]_\times + (1-\cos\theta)[\hat{\mathbf{w}}]_\times^2, \tag{1.17}$$

où $[\hat{\mathbf{w}}]_\times$ est la matrice antisymétrique qui correspond au vecteur untitaire $\hat{\mathbf{w}} = \frac{\mathbf{w}}{\|\mathbf{w}\|}$. A première vue, cette formulation peut sembler non définie lorsque $\theta = \|\mathbf{w}\|$ tend vers zéro. Toutefois, ce n'est pas le cas car l'équation (1.17) peut être réécrite comme suit :

$$\mathsf{R}([\mathbf{w}]_\times) = \exp([\mathbf{w}]_\times) = \mathcal{I}_3 + \frac{\sin\theta}{\theta}[\mathbf{w}]_\times + \frac{(1-\cos\theta)}{\theta^2}[\mathbf{w}]_\times^2. \tag{1.18}$$

Cette équation ne possède pas de singularité pour les faibles valeurs de θ comme on peut le voir en remplaçant $\frac{\sin\theta}{\theta}$ et $\frac{(1-\cos\theta)}{\theta^2}$ par les deux premiers termes de leurs développements de Taylor.

Cette représentation a des singularités qui se manifestent lorsque $\|\mathbf{w}\| = 2n\pi$ avec $n \geq 1$. Dans ce cas, il n'y a effectivement pas de rotation quelle que soit la

direction de $\mathbf{w}$. Toutefois, en pratique, cela peut être facilement évité comme suit : lorsque $\|\mathbf{w}\|$ devient trop proche de 2π, $\mathbf{w}$ peut être remplacé par $\left(1 - \frac{2\pi}{\|\mathbf{w}\|}\right)\mathbf{w}$, qui représente la même rotation lorsque la norme est plus petite que π. Autrement dit, une rotation de θ radians par rapport à $\mathbf{w}$ est équivalente à une rotation de $2\pi - \theta$ radians par rapport à $-\mathbf{w}$.

En résumé, la représentation axe/angle utilisant l'*exponential map* représente une rotation comme un vecteur de taille 3×1 qui comprend les valeurs de son axe et de son amplitude. Cette représentation évite le problème du blocage de cardan que l'on retrouve avec les angles d'Euler et ne nécessite pas de contrainte supplémentaire telle que celle nécessaire pour préserver la norme d'un quaternion. La matrice de rotation correspondante au vecteur peut être calculée avec l'équation (1.18). C'est donc une formulation tout à fait appropriée pour les méthodes de localisation de caméra que l'on abordera par la suite.

Linéarisation autour des petites rotations. Dans les applications de suivi 3D, le déplacement de la caméra entre deux images consécutives peut souvent être supposé comme étant faible, ainsi que les angles de rotation correspondants. Dans ce cas, il est possible d'utiliser une approximation au premier ordre de la rotation, qui linéarise le problème d'estimation de la pose et simplifie les calculs. Soit $\mathbf{Q}'$ la position du point 3D et $\mathbf{Q}$ sa nouvelle position après une faible rotation R par rapport à l'origine. Dans ce cas, toutes les formulations décrites ci-dessus aboutissent à la même approximation au premier ordre.

$$\begin{aligned} \mathbf{Q}' &= \mathsf{R}\,\mathbf{Q} \\ &\approx (\mathcal{I}_3 + [\mathbf{w}]_\times)\,\mathbf{Q} \\ &= \mathbf{Q} + [\mathbf{w}]_\times \mathbf{Q}, \end{aligned} \tag{1.19}$$

où la matrice $[\mathbf{w}]_\times$ est la matrice antisymétrique de l'équation (1.16).

1.2 Techniques d'estimation

1.2.1 Techniques de minimisation de moindres carrés

En vision par ordinateur en particulier, un des problèmes à résoudre peut être vu comme la recherche d'un ensemble de paramètres $\mathbf{x} = \mathbf{x}_1, ..., \mathbf{x}_n$ tel que :

$$f(\mathbf{x}) = \mathbf{y}, \tag{1.20}$$

où f est la fonction modélisant le problème étudié et $\mathbf{y} = \mathbf{y}_1, ..., \mathbf{y}_p$ un ensemble de mesures. Dans la pratique, cette égalité stricte ne peut pas être obtenue. Ceci est dû aux erreurs de mesure et de calcul numérique par exemple. On définit dans ce cas l'erreur résiduelle comme étant la différence entre les mesures et le modèle appliqué aux paramètres estimés :

$$\mathbf{r}(\mathbf{x}) = f(\mathbf{x}) - \mathbf{y}. \tag{1.21}$$

L'approche couramment utilisée pour résoudre le problème posé est alors la méthode des moindres carrés. Elle est utilisée dans les cas de problèmes sur-contraints, où l'on dispose de plus de données que d'inconnues. Cela revient à trouver les paramètres qui vérifient l'équation :

$$\hat{\mathbf{x}} = argmin_{\mathbf{x}\in\mathbb{R}^n}\, \mathcal{E}(\mathbf{x}), \tag{1.22}$$

où la fonction de coût $\mathcal{E}$ à minimiser est définie par :

$$\begin{array}{rcl} \mathcal{E}(\mathbf{x}) & = & \frac{1}{2}\left\|f(\mathbf{x}) - \mathbf{y}\right\|^2 \\ & = & \frac{1}{2}\left\|\mathbf{r}(\mathbf{x})\right\|^2. \end{array} \tag{1.23}$$

En particulier, dans le cas où la distribution des erreurs est gaussienne, l'estimation aux moindres carrés correspond à l'estimation du maximum de vraisemblance. Dans ce cas, la solution obtenue est optimale au sens statistique du terme.

L'approche utilisée pour résoudre ce problème dépend alors de la linéarité de la fonction $f(\mathbf{x})$. Avant de présenter les techniques itératives résolvant les problèmes de moindres carrés non-linéaires, nous considérons tout d'abord le cas où la fonction $f(\mathbf{x})$ est linéaire.

1.2.1.1 Moindres carrés linéaires

Lorsque la fonction f est linéaire, il existe une matrice A telle que pour tout $\mathbf{x}$, $f(\mathbf{x}) = \mathsf{A}\mathbf{x}$. Deux cas de figure sont alors à différencier.

Système linéaire homogène. Lorsque le vecteur des mesures $\mathbf{y}$ est nul, on parle de système homogène. La résolution au sens des moindres carrés minimise le critère suivant :

$$\mathcal{E}(\mathbf{x}) \;=\; \left\|\mathsf{A}\mathbf{x}\right\|^2.$$

La solution aux moindres carrés minimisant $\mathcal{E}$ sous la contrainte $\|\mathbf{x}\| = 1$ est donnée par le vecteur propre associé à la plus petite valeur propre de la matrice A. Ce vecteur propre peut être facilement obtenu en utilisant la décomposition SVD (*Singular Value Decomposition*), de A :

$$\mathsf{A} \;=\; \mathsf{U}\Sigma\mathsf{V}^{\mathsf{T}}.$$

La solution recherchée est donnée par la colonne de V associée à la plus petite valeur propre, en pratique, la dernière colonne de V.

Système linéaire non-homogène. Dans le cas où le vecteur des mesures $\mathbf{y}$ est non-nul. Un système non homogène $\mathsf{A}\mathbf{x} = \mathbf{y}$ a une solution unique lorsque le rang de A est plein. la solution au sens des moindres carrés minimise le critère d'erreur suivant :

$$\mathcal{E}(\mathbf{x}) \quad = \quad \|\mathsf{A}\mathbf{x} - \mathbf{y}\|^2 .$$

Elle est donnée par la résolution des équations normales induites par le système :

$$\begin{aligned} \mathsf{A}^\mathsf{T}\mathsf{A}\mathbf{x} &= \mathsf{A}^\mathsf{T}\mathbf{y} \\ \mathbf{x} &= \underbrace{(\mathsf{A}^\mathsf{T}\mathsf{A})^{-1}\mathsf{A}^\mathsf{T}}_{\mathsf{A}^+}\mathbf{y}, \end{aligned}$$

où A^+ est appelée une pseudo-inverse de A.

1.2.1.2 Méthodes de résolution non linéaires

Lorsque la fonction f est non linéaire, il est possible de résoudre le problème posé en utilisant une méthode itérative. L'hypothèse faite par cette famille de méthodes est que la fonction est localement linéaire. Le principe est alors de trouver la direction et la longueur de pas (c'est à dire la distance à parcourir dans cette direction), dans l'espace des paramètres, qui permet de diminuer au mieux l'erreur résiduelle. Les paramètres sont alors modifiés à l'aide de l'incrément ainsi calculé et le processus est réitéré depuis la nouvelle valeur des paramètres.

De manière plus formelle, on cherche itérativement le vecteur de paramètres $\hat{\mathbf{x}}$ pour lequel l'erreur $\|\mathbf{r}\| = \|f(\hat{\mathbf{x}}) - \mathbf{y}\|$ est minimale. C'est un problème de moindres carrés non linéaires et la recherche de $\hat{\mathbf{x}}$ dans l'équation (1.22) est itérative en faisant l'hypothèse que f est localement linéaire. A chaque itération k, on calcule le pas (ou incrément) $\delta_k \in \mathbb{R}^N$ à appliquer au vecteur de paramètres $\mathbf{x}_k$ tel que $\mathbf{x}_{k+1} = \mathbf{x}_k + \delta_k$ en vérifiant que $||f(\mathbf{x}_{k+1}) - \mathbf{y}|| < ||f(\mathbf{x}_k) - \mathbf{y}||$ ce qui assure la décroissance de l'erreur. En partant de l'estimation initiale $\mathbf{x}_0$, on obtient une série $\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_n$ qui converge vers un minimum local $\hat{\mathbf{x}}$ de f.

Pour chaque δ_k , on a $\delta_k = \alpha_k D_k$ où D_k est la direction de descente et α_k la longueur du pas pour l'itération k. Il existe plusieurs stratégies dans le choix de la direction de descente D_k au point $\mathbf{x}_k$. Les méthodes de résolution non linéaires se distinguent sur la stratégie du choix de la direction et de la longueur de pas. Voici les méthodes principalement utilisées en vision par ordinateur :

Descente de gradient. La descente de gradient est une méthode de résolution du premier ordre. La direction de déplacement choisie est directement liée au gradient de la fonction de coût. Le principe général de la méthode de descente de gradient est de choisir à chaque itération un déplacement δ_{DG} dans la direction opposée au gradient $\mathsf{J}^\mathsf{T}\mathbf{r}$ de la fonction de coût.

$$\delta_{DG} = -\mathsf{J}^{\mathsf{T}}\mathbf{r} \tag{1.24}$$

Cette méthode converge dans tous les cas, même lorsque la solution initiale est éloignée du minimum recherché. L'avantage de cette approche est qu'elle permet une convergence efficace lorsque le critère à minimiser est convexe. La longueur de pas est généralement fixée à 1 car, en pratique, on observe souvent que $D = g$ est une bonne direction de descente loin de la solution $\hat{\mathbf{x}}$ mais qu'elle est à éviter dès que l'on entre dans le voisinage de cette solution, là où les termes du second ordre d'un développement de Taylor de f autour de $\hat{\mathbf{x}}$ jouent un grand rôle. En fait, le défaut de cette méthode de descente est d'ignorer la courbure de f en $\mathbf{x}$, qui est décrite par sa matrice Hessienne. Cette technique de minimisation est l'une des plus simples et souffre de quelques problèmes de rapidité de convergence notamment près du minimum de la fonction coût.

L'algorithme de Gauss-Newton. A l'inverse de l'algorithme de descente de gradient, la méthode d'optimisation non-linéaire de Gauss-Newton est plus sensible à la condition initiale, et ne converge pas forcément. Elle assure néanmoins une convergence plus efficace lorsque les paramètres initiaux sont proches de la solution.

Afin d'obtenir la direction et la longueur de l'incrément à chaque itération, cette méthode du second ordre s'appuie principalement sur une approximation quadratique locale de la fonction de coût, formulée en termes des matrices Jacobienne $\mathsf{J} = \frac{\partial \mathbf{r}}{\partial \delta}$ et Hessienne $\frac{\partial^2 \mathbf{r}}{\partial^2 \delta}$. L'évaluation de la matrice Hessienne est souvent coûteuse. L'approximation de Gauss-Newton permet d'écrire cette matrice : $\mathsf{N} = \mathsf{J}^{\mathsf{T}}\mathsf{J}$. Cette approximation est valide soit lorsque la fonction de coût est « proche » d'être linéaire, soit lorsque les résidus sont petits, ce qui est le cas dans les problèmes que l'on va traiter par la suite. L'approximation de Gauss-Newton confère une économie du calcul des dérivées secondes de la fonction de coût. Cette approximation $e(\delta)$ quadratique s'écrit :

$$e(\delta) = \frac{1}{2}\left\|\mathbf{r} + \mathsf{J}\delta\right\|^2 \tag{1.25}$$

$$e(\delta) = \frac{1}{2}(\mathbf{r} + \mathsf{J}\delta)^{\mathsf{T}}(\mathbf{r} + \mathsf{J}\delta) \tag{1.26}$$

$$e(\delta) = \frac{1}{2}\mathbf{r}^{\mathsf{T}}\mathbf{r} + \mathsf{J}\mathbf{r}\delta^{\mathsf{T}} + \frac{1}{2}\delta^{\mathsf{T}}\mathsf{J}^{\mathsf{T}}\mathsf{J}\delta. \tag{1.27}$$

Pour chaque itération de l'algorithme de Gauss-Newton, nous cherchons alors le déplacement qui minimise l'approximation courante de la fonction de coût :

$$\delta = argmin_{\delta}\, e(\delta). \tag{1.28}$$

Il s'agit d'un problème quadratique pour lequel un point stationnaire peut être estimé par l'annulation de la dérivée de l'approximation $e(\delta)$ par rapport au déplacement δ, ce qui donne les équations normales suivantes :

$$\frac{\partial e(\delta)}{\partial \delta} = \mathsf{J}^\mathsf{T}\mathbf{r} + \mathsf{J}^\mathsf{T}\mathsf{J}\delta = 0. \tag{1.29}$$

Cette équation peut se réécrire sous la forme suivante :

$$\mathsf{J}^\mathsf{T}\mathsf{J}\delta = -\mathsf{J}^\mathsf{T}\mathbf{r}. \tag{1.30}$$

L'algorithme de Gauss-Newton consiste donc à calculer la solution suivante pour δ_{GN}, avec $\mathbf{g} = \mathsf{J}^\mathsf{T}\mathbf{r}$ le vecteur gradient de la fonction de coût :

$$\delta_{GN} = -\mathsf{N}^{-1}\mathbf{g}. \tag{1.31}$$

On calcule alors les itérations successives de Gauss-Newton jusqu'à convergence. Malheureusement, il arrive que le minimum obtenu soit un minimum local différent de la solution recherchée $\hat{\mathbf{x}}$ ou même que l'algorithme diverge. Le comportement et la convergence dépendent fortement de l'estimation initiale $\mathbf{x}_0$.

L'algorithme de Levenberg-Marquardt. La méthode d'optimisation non-linéaire de Levenberg-Marquardt (Levenberg (1944); Marquardt (1963)) est devenue un standard parmi les méthodes itératives d'optimisation non-linéaire. Cet algorithme est une variante de Gauss-Newton consistant à effectuer une régularisation des équations normales. La matrice N est remplacée par une matrice régularisée $\mathsf{N}' = \mathsf{N} + \mathsf{W}(\lambda)$, où $\lambda \in \mathbb{R}$ et $\lambda > 0$. La matrice $\mathsf{W}(\lambda)$ est appelée matrice de régularisation. Les choix suivants ont été proposés pour $\mathsf{W}(\lambda)$:

- ▷ $\mathsf{W}(\lambda) = \lambda\mathsf{I}$. C'est le choix original de Levenberg et Marquardt (Levenberg (1944); Marquardt (1963)) repris par Triggs *et al.* (2000).
- ▷ $\mathsf{W}(\lambda) = (1+\lambda)diag(\mathsf{N})$, c'est-à-dire multiplication des coefficients diagonaux de N par $(1+\lambda)$. Ce choix est dû à Seber et Wild (1989).

Dans l'algorithme de Levenberg-Marquardt, on résout donc une équation similaire à l'équation (1.31). Si on considère le cas où les termes diagonaux de la matrice $\mathsf{N} = \mathsf{J}^\mathsf{T}\mathsf{J}$ sont multipliés par $(1+\lambda)$. Cela revient à résoudre :

$$\delta_{LM} = -\mathsf{N}'^{-1}\mathbf{g}, \tag{1.32}$$

avec $\mathsf{N}'(i,i) = (1+\lambda)\mathsf{N}(i,i)$, $\forall i$ et $\mathsf{N}'(i,j) = \mathsf{N}(i,j)$ pour $i \neq j$.

Le fait de modifier ainsi la diagonale de la matrice est appelé amortissement (*damping* en anglais), et le paramètre λ est dit coefficient d'amortissement. Ce paramètre permet le contrôle de la régularisation. Lourakis et Argyros (2005) proposent, une heuristique pour initialiser et mettre à jour ce paramètre (typiquement $\lambda = 10^{-3}$).

- ▷ Si l'incrément δ_{LM} obtenu par la résolution de l'équation (1.32) permet une diminution de l'erreur, alors δ_{LM} est accepté et λ est divisé par une constante, souvent 10, avant la prochaine itération.
- ▷ Au contraire, si δ_{LM} entraîne une augmentation de l'erreur, alors λ est multiplié par la constante, on met à jour N' et on résout à nouveau (1.32) jusqu'à obtenir un δ_{LM} qui fait décroître l'erreur.

Si le coefficient λ est grand, alors la matrice N' de l'équation (1.32) est proche d'une matrice diagonale et le pas δ_{LM} correspond presque à une descente de gradient. Par contre, si λ est petit, le système à résoudre (1.32) se rapproche de (1.31) et on obtient un pas proche de celui de Gauss-Newton.

La même interprétation peut être faite pour le cas où la matrice de régularisation est $W(\lambda) = \lambda\mathcal{I}$. Lorsque λ est petit, $\mathsf{N} = \mathsf{N} + \lambda\mathcal{I} \approx \mathsf{N}$, on obtient une itération de type Gauss-Newton, donnée par l'équation (1.31). Lorsque λ est grand, $\mathsf{N} = \mathsf{N} + \lambda\mathcal{I} \approx \mathcal{I}$, on obtient alors une solution proche de $\delta = -\mathbf{g}$, qui correspond à une itération du type descente de gradient (1.24).

L'algorithme est adaptatif et contrôle automatiquement l'amortissement. De ce fait, il est capable d'alterner entre une lente descente de gradient loin du minimum local et une rapide convergence quadratique dans son voisinage. Chaque itération combine la méthode de Gauss-Newton et la méthode de descente de gradient afin de profiter de leur avantage respectif. Ainsi, lorsque la solution est loin du minimum local, on privilégie la descente de gradient qui offre une convergence lente mais assurée. Au contraire, quand on s'approche du minimum local, c'est la méthode de Gauss-Newton qui est mise en avant et permet une convergence plus efficace.

A noter que ces méthodes itératives sont particulièrement sensibles aux minimums locaux et n'assurent pas la convergence vers le minimum global de la fonction de coût. Cela implique que la condition initiale doit être aussi proche que possible de la solution recherchée.

1.2.2 Estimation robuste

Quelle que soit la méthode de résolution non linéaire choisie, la solution doit composer avec le problème de robustesse au bruit dans les données. En vision par ordinateur ce bruit peut venir de l'acquisition de données réelles, des occultations, des changements d'éclairages, *etc*. Il est plus efficace de supposer que les données sont erronées et d'utiliser un processus d'estimation robuste pour régler ce problème. Dans la littérature liée aux méthodes statistiques, de nombreuses approches existent pour traiter ces sources d'erreur. Parmi les algorithmes de rejet des valeurs aberrantes, les méthodes de vision par ordinateur ont inclus la Transformée de Hough et RANSAC Fischler et Bolles (1981). Ces approches traitent l'écart-type des données considérées correctes (*inliers*) comme une constante qui doit être estimée. D'autre part, les méthodes statistiques telles que Least Median Squares (LMedS) et les M-estimateurs ont été développées et utilisées en vision par ordina-

teur. La plupart de ces approches ont été utilisées pour estimer la pose ou le déplacement de la caméra (voir, par exemple, Fischler et Bolles (1981); Haralick *et al.* (1989); Kumar et Hanson (1994); Simon et Berger (2002); Wang et Suter (2004)). Le lecteur est renvoyé à Stewart (1999) pour une revue des différentes techniques d'estimation robustes appliquées en vision par ordinateur.

1.2.2.1 M-Estimateur

Alors que la méthode classique des moindres carrés fait l'hypothèse que le bruit sur les mesures suit une distribution normale (gaussienne), une version robuste des moindres carrés est nécessaire quand il y a des valeurs aberrantes parmi les mesures. Dans ce cas, il est préférable d'utiliser un M-estimateur Huber (1981); Hampel *et al.* (1986); Stewart (1999), qui consiste à appliquer une fonction de pénalité robuste $\rho(\mathbf{r})$ aux résidus :

$$\mathcal{E}_{RLS}(\mathbf{x}) = \sum_i \rho\left(\|\mathbf{r}_i(\mathbf{x})\|\right), \tag{1.33}$$

au lieu de les élever au carré. La fonction de coût est noté $\mathcal{E}_{RLS}$, où RLS fait référence à *Robust Least Squares*. Nous pouvons prendre la dérivée de cette fonction par rapport à $\mathbf{x}$ et la mettre à 0,

$$\sum_i \psi(\|\mathbf{r}_i\|)\frac{\partial\|\mathbf{r}_i\|}{\partial\mathbf{x}} = \sum_i \frac{\psi(\|\mathbf{r}_i\|)}{\|\mathbf{r}_i\|}{\mathbf{r}_i}^\top\frac{\partial\mathbf{r}_i}{\partial\mathbf{x}} = 0, \tag{1.34}$$

où $\psi(\mathbf{r}) = \rho'(\mathbf{r})$ est la dérivée de $\rho(\mathbf{r})$ et est appelée la fonction d'influence. Si nous introduisons une fonction de poids, $w(\mathbf{r}) = \Psi(\mathbf{r})/\mathbf{r}$, on peut observer que trouver le point stationnaire de (1.33) en utilisant (1.34) est équivalent à minimiser le problème des moindres carrés itérativement repondérés (IRLS pour *Iteratively Reweighted Least Squares*)

$$\mathcal{E}_{IRLS} = \sum_i w(\|\mathbf{r}_i\|)\,\|\mathbf{r}_i\|^2, \tag{1.35}$$

où $w(\|\mathbf{r}_i\|)$ joue un rôle de pondération locale. L'algorithme IRLS alterne entre le calcul des fonctions de poids $w(\|\mathbf{r}_i\|)$ et la résolution du problème des moindres carrés pondérés résultant (avec des valeurs de w fixés).

Plusieurs M-estimateurs ont été proposés dans la littérature (Huber (1981)). Les M-estimateurs se différencient par leur gestion des résidus aberrants. Les M-estimateurs de Huber et Tuckey sont parmi les plus employés dans la communauté de vision par ordinateur. Pour ces M-estimateurs, la contribution quadratique des erreurs devient :

▷ Pour le M-estimateur de Huber, l'évolution du poids des points aberrants est linéaire avec leur résidu :

$$\rho(x) = \begin{cases} \frac{x^2}{2} & \text{si } |x| \leq \sigma \\ \sigma\left(|x| - \frac{\sigma}{2}\right) & \text{sinon} \end{cases} \tag{1.36}$$

▷ Le M-estimateur de Tukey rend l'influence des résidus constante pour tous les points aberrants :

$$\rho(x) = \begin{cases} \frac{\sigma^2}{2}\left(1 - \left(1 - \left(\frac{x}{c}\right)^2\right)^3\right) & \text{si } |x| \leq \sigma \\ \frac{c^2}{6} & \text{sinon} \end{cases} \tag{1.37}$$

Pour tous les M-estimateurs, il est nécessaire de fixer le seuil σ à partir duquel les valeurs de résidus sont considérées aberrantes. En pratique il est possible dans certains problèmes de fixer ce seuil empiriquement, mais il peut être automatiquement estimé à partir des résidus mesurés. Par exemple, la médiane des écarts absolus à la médiane (MAD pour *Median Absolute Deviation*) permet d'estimer ce seuil dans les cas où la distribution des résidus étudiés peut être assimilée à une distribution gaussienne (Malis et Marchand (2006)).

1.2.2.2 RANSAC et LMedS

Les M-estimateurs peuvent donc aider à réduire l'influence des valeurs aberrantes. Cependant dans certains cas, initialiser avec un trop grand nombre de valeurs aberrantes peut empêcher l'IRLS (ou d'autres algorithmes de minimisation non-linéaire) de converger vers l'optimum global. Une meilleure approche est souvent de trouver un ensemble de correspondances *inlier* pour initialiser le problème. Deux méthodes largement utilisées dans ce cas sont RANSAC (pour *RANdom SAmple Consensus*, (Fischler et Bolles (1981)), et LMedS(pour *Least Median of Squares*, (Rousseeuw et Leroy (1987)). Le principe est de sélectionner un sous-ensemble des M mesures $\mathbf{y}$ qui offre la meilleure estimation des paramètres du modèle $\hat{\mathbf{x}}$. Ce sous-ensemble est de taille minimale N c'est-à-dire supérieure ou égale au nombre minimum de mesures nécessaires à la résolution du problème posé. Ces deux méthodes varient selon leur critère d'optimalité.

Critère d'optimalité pour RANSAC. Le critère de l'optimalité de $\mathbf{x}$ correspondant à la mesure de qualité est donné par la taille du support associé au jeu de paramètres $\hat{\mathbf{x}}_i$. Le support est l'ensemble des mesures, parmi les M mesures $\mathbf{y}$, qui sont satisfaites par les paramètres, c'est-à-dire pour lesquelles $\|f(\hat{\mathbf{x}}_i) - \mathbf{y}_i\| < \xi$ où ξ est un seuil à définir. La valeur ξ dépend de l'application mais elle est souvent aux alentours de 1 à 3 pixels en vision par ordinateur. Le tirage finalement retenu est celui qui donne le support le plus grand.

Critère d'optimalité pour LMedS. Alors que l'algorithme de RANSAC vise à maximiser la taille du support, c'est-à-dire le nombre de points non-aberrants, la méthode LMedS a pour but de minimiser l'erreur médiane des résidus. L'approche LMedS permet de s'affranchir du seuil ξ lorsque le taux de points non-aberrants est connu et est supérieur à 50%. Le critère d'optimalité pour le LMedS dépend de la valeur médiane de $\|f(\hat{\mathbf{x}}_i) - \mathbf{y}_i\|^2$. La sélection du meilleur tirage se fait en choisissant celui qui donne l'erreur la plus faible.

Plus précisément, RANSAC et LMedS se composent, pour chacune des itérations $i = 1, .., S$, des étapes suivantes :

- ▷ **Tirage.** Réalisation d'un tirage aléatoire d'un ensemble, noté E_i, de N mesures parmi les M mesures $\mathbf{y}$ du problème de départ.
- ▷ **Résolution.** Résolution de l'équation décrivant le problème étudié à l'aide d'une des techniques d'optimisation par les moindres carrés pour obtenir une solution $\hat{\mathbf{x}}_i$ à partir des données sélectionnées E_i.
- ▷ **Mesure de qualité.** Mesure de la qualité de l'estimation des paramètres $\hat{\mathbf{x}}_i$. Ce critère dépend de la méthode choisie (RANSAC ou LMedS).

Le tirage $\hat{i}$, qui optimise la mesure de qualité définie est alors conservé. Les paramètres $\hat{\mathbf{x}}$ sont alors calculés à partir de toutes les mesures qui respectent le critère de qualité pour la valeur $\hat{\mathbf{x}}_{\hat{i}}$ des paramètres correspondant au tirage $\hat{i}$. Ces mesures, considérées non-aberrantes, sont généralement appelées *inliers*.

Lorsque le nombre de mesures est trop grand, afin de diminuer les temps de calcul, tous les sous-ensembles de N mesures ne sont pas testés. Cette modification de RANSAC, qui peut augmenter de manière significative ces performances, est appelée Preemptive RANSAC (Nistér (2003)). Notons qu'il existe d'autres variantes de la méthode RANSAC (PROSAC, MLESAC, ARRSAC, etc.), dont une évaluation est proposée dans Raguram *et al.* (2008).

1.3 Localisation et reconstruction 3D par vision

Nous présentons dans cette section les algorithmes de base permettant la localisation d'une caméra et la reconstruction 3D de la scène observée sous forme d'un nuage de points. Nous introduisons tout particulièrement les notions de détection, description et appariement de points d'intérêt, de triangulation de points, de calcul de pose de caméra, *etc*.

1.3.1 Mise en correspondance 2D

Pour pouvoir localiser une caméra dans un environnement inconnu, il faut tout d'abord détecter des éléments fixes que l'on pourra facilement suivre dans les différentes images de la séquence. Ces éléments fixes, appelés éléments d'intérêt, seront dans cette thèse constitués de points 3D car ils sont omniprésents dans les types d'environnements qui nous intéressent.

Nous présentons dans cette section les différentes étapes permettant de détecter, de décrire et d'apparier les points d'intérêt 2D correspondants aux observations des points 3D dans les images. À la fin de ces trois étapes, nous obtenons un graphe de correspondances de points 2D qui sera utilisé par l'une des méthodes de *Structure from Motion* présentées dans la suite de ce chapitre, afin de localiser les images et de reconstruire des points 3D de la scène.

1.3.1.1 Détection de points d'intérêt

Nous nous intéressons ici à la détection de points 2D dit d'intérêt dans les images. Ces points 2D devant être facilement localisables et reconnaissables dans les différentes images, il est courant de prendre les zones saillantes ou les coins (*corner*) de l'image. Les détecteurs de points d'intérêt doivent être, de plus, suffisamment robustes aux changements de perspective de la scène intervenant avec le déplacement de la caméra, afin de pouvoir détecter les mêmes points dans toutes les images. Il y a là une notion d'invariance de transformation 2D, par exemple l'invariance à la transformation affine (Mikolajczyk et Schmid (2002); Lowe (2004); Tuytelaars et Mikolajczyk (2008)). Pour cela, nous employons dans cette thèse (concernée par le temps réel) les points d'intérêt fournis par le détecteur de HARRIS (Harris et Stephens (1988)).

Dans cette thèse, nous ne sélectionnons qu'une partie des points d'intérêt afin de gagner en temps de calcul. Pour cela nous sélectionnons un nombre fixe de points d'intérêt, les points les plus saillants grâce à l'indice de discrimination délivré par le détecteur utilisé. De plus, nous utilisons un système de baquet qui découpe l'image en sous-zones et nous cherchons les points d'intérêt dans ces sous-zones. Ceci permet de garantir une certaine répartition des points d'intérêt dans l'image, ce qui améliore la qualité des résultats (Zhang (1998)).

1.3.1.2 Description de points d'intérêt

Afin de retrouver les observations d'un même point 3D dans une séquence d'images, il est nécessaire d'identifier le point de manière unique. Pour cela, nous employons un système de signature sur chaque point 3D. Cette signature est constituée d'un ensemble de caractéristiques, les descripteurs locaux, et permet de mesurer la similarité entre deux points. Ces descripteurs doivent être suffisamment discriminants pour pouvoir différencier des points 3D, mais aussi suffisamment génériques pour accepter des variations entraînées soit par la scène (notamment variations d'éclairage) soit par le mouvement de la caméra (changements de point de vue, variation d'échelle, *etc.*).

La première grande catégorie de descripteurs utilise des histogrammes de gradients. Le plus connu et le fondateur du genre est SIFT pour *Scale invariant feature transform* (Lowe (2004)). Un des descripteurs les plus utilisés aujourd'hui est SURF (Bay *et al.* (2006)). Il est fortement inspiré de SIFT, mais utilise des ondelettes de

Haar 2D plutôt que des gradients. En comparaison, le descripteur SURF (Bay *et al.* (2006)) est beaucoup plus rapide (mais moins robuste que SIFT) et peut donc être employé dans des applications temps réel. Il existe une version temps réel du descripteur SIFT basée sur une implémentation sur GPU (Sinha *et al.* (2006)). Dans cette thèse nous utilisons principalement le descripteur SURF qui offre le meilleur rapport qualité/temps de traitement pour nos applications.

A noter qu'une nouvelle approche de descripteurs est apparue dans la littérature. Il s'agit de descripteurs binaires qui sont plus rapides que les descripteurs cités précédemment. Il y a tout d'abord BRIEF (Calonder *et al.* (2010)) qui n'est invariant ni par rotation ni par échelle. Une version invariante par rotation appelée ORB (pour *Oriented fast and Rotated BRIEF*) a ensuite été développée par Rublee *et al.* (2011). Similairement, Leutenegger *et al.* (2011) ont proposé BRISK (pour *Binary Robust Invariant Scalable Keypoints*), un descripteur binaire à la fois invariant par rotation et par changement d'échelle.

1.3.1.3 Mise en correspondance de points d'intérêt

Une fois que nous avons un ensemble de points d'intérêt avec leurs descripteurs associés, nous pouvons tenter de les mettre en correspondance : c'est l'étape d'appariement. L'objectif de cette étape est de retrouver les observations d'un même point 3D parmi l'ensemble des points d'intérêt détectés. La phase d'appariement proprement dite consiste à calculer un score de ressemblance entre chaque point d'intérêt à partir de leur descripteur, et de coupler les observations ayant les scores les plus importants. L'appariement doit de plus respecter la contrainte d'unicité : un point 3D ne doit avoir qu'une seule observation par image.

Cette étape peut être affectée par de mauvais appariements (*outliers*), ce qui donnera au final une reconstruction peu fidèle de la réalité. Pour éviter cela, et éliminer ces *outliers*, les méthodes robustes telles que RANSAC (voir la section 1.2.2.2) peuvent être utilisées avec, par exemple un critère sur la distance à la droite épipolaire utilisant les contraintes géométriques fournies par la matrice Fondamentale (voir la section 1.3.2.1).

1.3.2 Estimation du déplacement inter-caméra

Cette section traite de l'estimation du déplacement relatif entre deux images. Ce problème peut être résolu par différentes techniques (Longuet-Higgins (1987); Hartley (1995); Nistér (2004)). Nous commençons par rappeler les concepts de la géométrie épipolaire puis nous présentons l'estimation du déplacement entre deux images par la méthode des cinq points.

1.3.2.1 Géométrie épipolaire

La géométrie épipolaire décrit les contraintes reliant les observations par deux caméras d'une même scène, notées $\mathcal{C}_1$ et $\mathcal{C}_2$ (voir la figure 1.4). Ces contraintes sont directement liées au déplacement relatif entre les deux caméras, et ce de manière indépendante de la structure de la scène observée (voir notamment les livres de Faugeras (1993) et de Hartley et Zisserman (2004)).

La géométrie épipolaire résume les contraintes géométriques dues au fait que deux observations 2D $\mathbf{q}_1$ et $\mathbf{q}_2$ sont issues d'un même point 3D. Dans le cas d'une caméra perspective, il existe un faisceau de points 3D qui se projettent tous en un même point 2D. Si nous projetons ce faisceau dans la deuxième caméra, nous obtenons une droite 2D qui est la projection de tous les points 3D associés au point 2D de la première caméra. Ainsi, pour un point $\mathbf{q}_1$ de l'image de la première caméra, il est possible de calculer la droite $\mathbf{l}_2$ sur laquelle se situe l'observation correspondante dans la deuxième caméra (figure 1.4) :

$$\mathbf{l}_2 \sim \mathsf{F}\tilde{\mathbf{q}}_1. \tag{1.38}$$

La droite $\mathbf{l}_2$ est appelée droite épipolaire associée à $\mathbf{q}_1$. De plus, ces deux observations $\mathbf{q}_1$ et $\mathbf{q}_2$ correspondant au même point de l'espace, elles vérifient :

$$\tilde{\mathbf{q}}_2^{\mathsf{T}}\mathsf{F}\tilde{\mathbf{q}}_1 = 0. \tag{1.39}$$

La matrice F est appelée la matrice Fondamentale (Faugeras (1992); Hartley (1992)), c'est une matrice 3×3 de rang 2. Dans chacune des images, il y a un point particulier nommé épipole. Les épipoles $\mathbf{e}_1$ et $\mathbf{e}_2$ correspondent à la projection des centres optiques dans les deux images. Ces points forment le noyau de la matrice Fondamentale : $\mathsf{F}\tilde{\mathbf{e}}_1 = 0$ et $\mathsf{F}^{\mathsf{T}}\tilde{\mathbf{e}}_2 = 0$. Cette matrice peut se décomposer de la manière suivante :

$$\mathsf{F} \sim \mathsf{K}_2^{-\mathsf{T}}\mathsf{E}\mathsf{K}_1^{-1}, \tag{1.40}$$

avec K_1, K_2 les matrices des paramètres intrinsèques des deux caméras $\mathcal{C}_1$ et $\mathcal{C}_2$. $\mathsf{K}_2^{-\mathsf{T}}$ désigne la transposée inverse de K_2. La matrice E est appelée la matrice Essentielle, c'est une matrice 3×3 ayant deux valeurs propres égales et la troisième est nulle. La matrice Essentielle E peut être vue comme étant le cas particulier de la matrice Fondamentale dans le cas où les matrices de calibrage K_1 et K_2 des caméras sont connues. Dans ce cas, l'équation (1.39) s'écrit :

$$\tilde{\mathbf{q}}_2^{\mathsf{T}}\left(\mathsf{K}_2^{-\mathsf{T}}\mathsf{E}\mathsf{K}_1^{-1}\right)\tilde{\mathbf{q}}_1 = 0. \tag{1.41}$$

Extraire $\mathsf{R}_{1\to2}$ et $\mathbf{t}_{1\to2}$ à partir de E. A partir de la matrice E, il est possible d'extraire la rotation et la translation correspondant au déplacement relatif entre les caméras $\mathcal{C}_1$ et $\mathcal{C}_2$. En effet, la matrice Essentielle n'est composée que des paramètres extrinsèques du déplacement entre ces deux caméras. Il existe donc une re-

lation qui lie la matrice Essentielle E à ce déplacement relatif, définie par le couple $(\mathsf{R}_{1\to2}, \mathbf{t}_{1\to2})$ et elle s'écrit :

$$\mathsf{E} = [\mathbf{t}_{1\to2}]_\times \mathsf{R}_{1\to2}, \tag{1.42}$$

avec $\mathsf{R}_{1\to2}$ la matrice de rotation entre $\mathcal{C}_1$ et $\mathcal{C}_2$, $\mathbf{t}_{1\to2}$ le vecteur de translation entre C_1 et C_2, et $[\mathbf{t}]_\times$ est la matrice antisymétrique associée au vecteur $\mathbf{t} = [t_1, t_2, t_3]^\mathsf{T}$:

$$[\mathbf{t}]_\times = \begin{pmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{pmatrix}.$$

La matrice $\mathsf{R}_{1\to2}$ et le vecteur $\mathbf{t}_{1\to2}$ peuvent être retrouvés en décomposant la matrice E par SVD (Hartley et Zisserman (2004)). En général, il y a alors quatre solutions différentes de $(\mathsf{R}_{1\to2}, \mathbf{t}_{1\to2})$ pour une matrice Essentielle. Ensuite, le principe est de reconstruire un point par triangulation pour lever l'ambiguïté et trouver le couple correct $(\mathsf{R}_{1\to2}, \mathbf{t}_{1\to2})$. En effet, la solution recherchée est alors celle dont le point reconstruit est en face des deux caméras. Une décomposition efficace de E en R et $\mathbf{t}$ est décrite dans Nistér (2004).

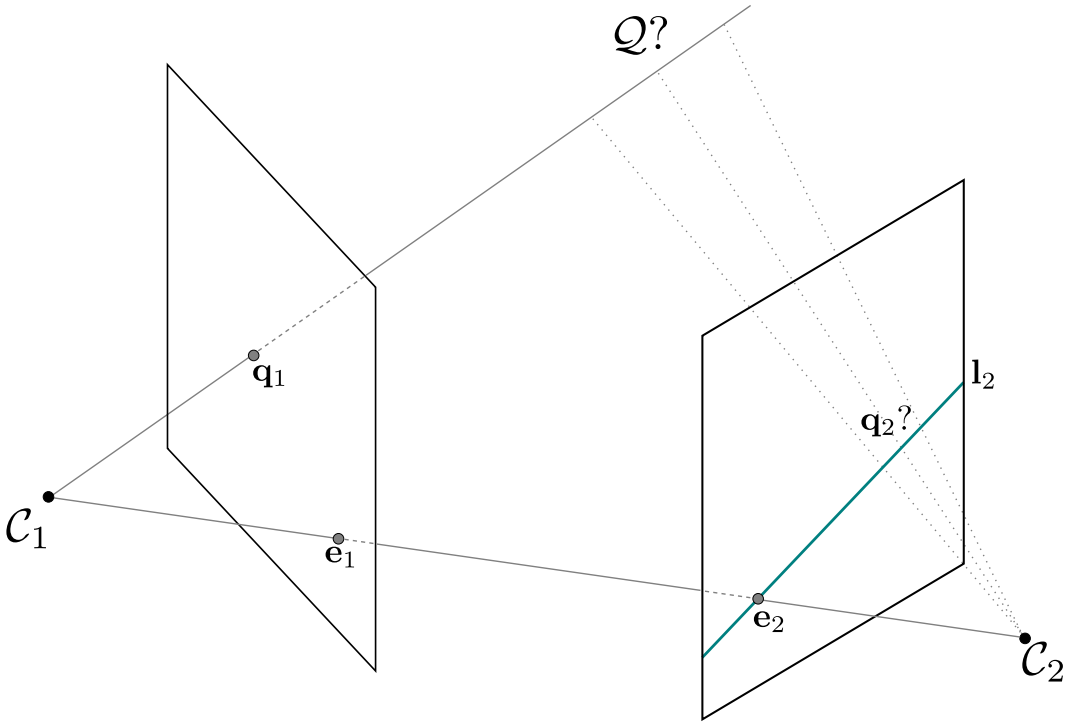


FIGURE 1.4 – **Géométrie épipolaire.** La géométrie épipolaire définit des contraintes géométriques entre les différentes observations d'un même point de l'espace.

1.3.2.2 Estimation de la matrice Fondamentale

Dans le cas où les paramètres intrinsèques des caméras sont inconnus, $\mathsf{R}_{1\to2}$ et $\mathbf{t}_{1\to2}$ sont calculés à partir de la matrice Fondamentale (Hartley et Zisserman (2004)). Dans ce cas, le déplacement inter-caméra ne peut être retrouvé qu'à une transformation projective près. En particulier, ceci induit qu'il est impossible de retrouver les rapports de distance et les angles. Dans ce cas le calcul de la transformation relative entre la caméra $\mathcal{C}_1$ et la caméra $\mathcal{C}_2$ se fait en estimant la matrice Fondamentale à partir de correspondances entre les deux images. Une solution simple et directe pour $n \geq 8$ points non coplanaires est l'algorithme de huit points de Longuet-Higgins (1981). A partir de cette estimation de la matrice Fondamentale, il est possible de retrouver les matrices de projection de chaque caméra (Hartley et Zisserman (2004)). En effet, la transformation entre les deux caméras est connue à une transformation projective de l'espace 3D près. Les matrices de calibrage K_1 et K_2 peuvent être retrouvées par une procédure appelée autocalibrage (Bougnoux (1998)). La reconstruction projective pourra alors être transformée en une reconstruction euclidienne. A noter que la solution de l'algorithme de huit points est dégénérée lorsque les points 3D, associés aux correspondances 2D-2D, sont coplanaires.

1.3.2.3 Estimation de la matrice Essentielle

Dans le cas où le calibrage des caméras est connu, c'est la matrice Essentielle E qui est utilisée pour diminuer les erreurs d'estimations. Elle est estimée à partir de correspondances 2D-2D en utilisant la contrainte épipolaire. La solution nécessite au minimum cinq mises en correspondance 2D-2D (Kruppa (1913)). Un algorithme efficace a été proposé pour ce problème par Nistér (2004). Puis $\mathsf{R}_{1\to2}$ et $\mathbf{t}_{1\to2}$ peuvent être estimés à partir de la matrice Essentielle par SVD. L'algorithme des cinq points proposé par Nister est devenu le standard pour l'estimation du déplacement inter-caméra à partir de mises en correspondance 2D-2D car c'est une méthode efficace et minimale. La translation sera alors connue à un facteur d'échelle près seulement. A noter qu'à l'inverse de l'algorithme de huit points, l'algorithme de cinq points fonctionne également pour des points coplanaires. Enfin, notons que l'algorithme de huit points fonctionne pour les cas où la caméra est calibrée ou non calibrée, alors que l'algorithme de cinq points suppose forcément que la caméra est calibrée.

1.3.2.4 Estimation du déplacement inter-caméra par l'observation d'un plan 3D

En géométrie projective, deux images non calibrées d'un plan 3D sont reliées par une homographie plane H, avec H une matrice (3×3) régulière. La matrice H est homogène : elle est définie à un facteur près et possède donc 8 degrés de liberté. Plus formellement, si $\mathbf{q}_1$ est la projection dans une vue d'un point du plan et $\mathbf{q}_2$

sa projection dans une deuxième vue, alors les deux projections sont liées par la transformation linéaire projective :

$$\tilde{\mathbf{q}}_2 \sim \mathsf{H}\tilde{\mathbf{q}}_1. \tag{1.43}$$

Nous montrons dans cette section comment résoudre le problème d'estimation du déplacement relatif entre deux caméras à partir des homographies H induites par l'observation d'un plan π de deux points de vue différents $(1, 2)$. Nous nous plaçons dans le contexte décrit par la figure 1.5 : les deux caméras (notées $\mathcal{C}_1, \mathcal{C}_2$) observent un plan du modèle. Soit R la matrice de rotation et $\mathbf{t}$ le vecteur translation de la caméra entre les deux vues, et soit $\mathbf{n}$ la normale et d la distance à l'origine du plan π, exprimées dans le repère de la première caméra. L'équation du plan π dans ce même repère est $(\mathbf{n}\ d)^\mathsf{T}$. L'homographie H induite par ce plan est :

$$\mathsf{H} = \mathsf{K}_2(\mathsf{R} - \frac{\mathbf{t}\mathbf{n}^\mathsf{T}}{d})\mathsf{K}_1^{-1}. \tag{1.44}$$

Notons que $\mathsf{K}_1, \mathsf{K}_2$ correspondent aux matrices des paramètres intrinsèques associées aux deux caméras $\mathcal{C}_1$ et $\mathcal{C}_2$. Dans le cas monoculaire ces deux matrices seraient les mêmes.

L'équation (1.44) est couramment utilisée pour estimer le déplacement inter-caméra. En effet, à partir de l'estimation de la matrice H (obtenue par exemple avec la méthode DLT détaillée dans les livres de Faugeras (1993) et de Hartley et Zisserman (2004)) il est possible d'extraire les 6 paramètres $(\mathsf{R}, \mathbf{t})$ du déplacement relatif. A noter qu'il y a une ambiguité entre la distance d et la norme de $\mathbf{t}$. Pour estimer le déplacement inter-caméra, il est alors nécessaire de connaître *a priori* la distance d.

1.3.3 Reconstruction 3D d'un nuage de points

La reconstruction d'un nuage de points 3D, aussi connue sous le nom de triangulation, consiste à déterminer les positions des points 3D à partir de leurs observations 2D par au moins deux caméras dont les poses sont connues. Le problème est simple dans le cas où il n'y a pas de bruit. Il suffit de trouver l'intersection des deux rayons rétroprojetés définis par chaque couple (caméra, point 2D). En pratique, à cause des bruits sur les différentes données (calibrage, pose des caméras, position des observations, *etc.*), les rayons ne s'intersectent pas. Plusieurs méthodes ont été proposées pour trouver « la meilleure solution » à ce problème. Nous citons ci-dessous les trois plus connues en vision.

- ▷ La méthode du point milieu est la plus rapide. Elle consiste à choisir comme solution à ce problème le point équidistant des deux rayons (voir la figure 1.6). Elle peut être étendue à plus de deux vues en calculant la triangulation

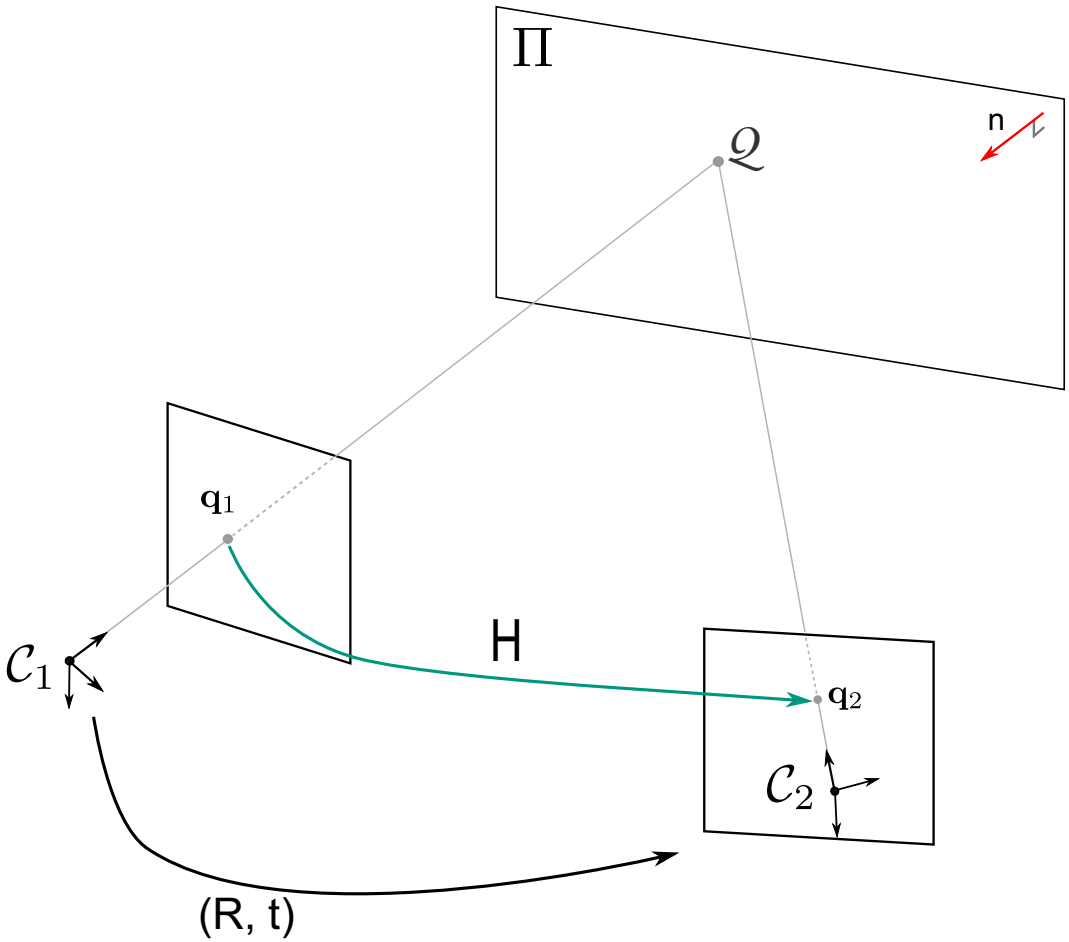


FIGURE 1.5 – **Homographies 2D.** Les coordonnées des observations correspondantes d'un point 3D situé sur un plan sont reliées par une homographie 2D.

par paire de caméras. Le résultat final de la triangulation est alors l'isobarycentre de ces points obtenus pour chaque paire de caméras ;

- ▷ Une méthode linéaire permet de trianguler un point observé par N-vues en utilisant la méthode DLT (Hartley et Zisserman (2004)). Elle a l'avantage d'être linéaire et peut s'étendre facilement à $N \geq 2$ observations. Elle conduit à la résolution d'un système d'équations linéaires ;
- ▷ Une méthode optimale dans le cas de bruit gaussien (Hartley et Sturm (1997)), mais a pour inconvénient de devoir trouver les racines d'un polynôme de degré 6.

Les détails, ainsi qu'une comparaison de ces trois méthodes, peuvent être trouvés dans Hartley et Sturm (1997). Récemment d'autres méthodes optimales de triangulation plus rapides ont été proposées par Kanatani *et al.* (2008); Lindstrom (2010).

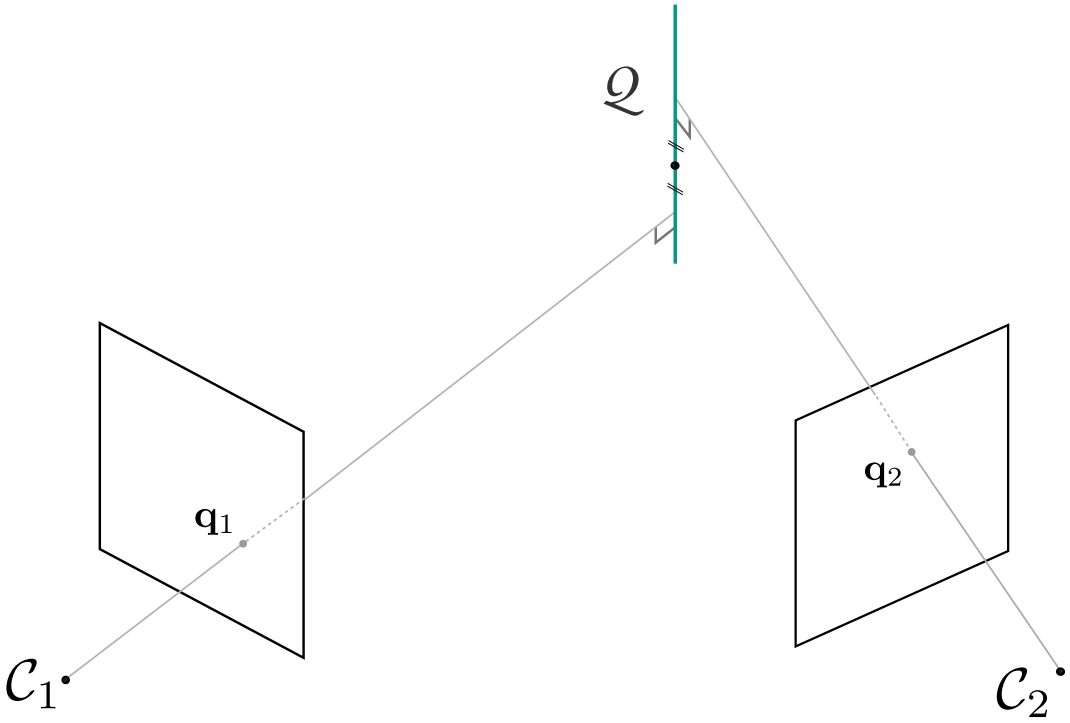


FIGURE 1.6 – **Triangulation de points 3D.** La structure de l'environnement peut être obtenue par triangulation des observations dans les images.

1.3.4 Estimation d'une pose de caméra

Nous cherchons cette fois-ci à déterminer la pose de la caméra à partir de correspondances entre les points 3D de positions connues et des points 2D détectés dans l'image courante. Cette approche permet de localiser la caméra dans le repère défini par ces points 3D. Pour cela, il est nécessaire de connaître la position d'au moins trois points 3D dans le cas calibré et six dans le cas non calibré. De nombreuses méthodes ont été proposées pour résoudre ce problème. On peut trouver une comparaison des méthodes de base dans Haralick *et al.* (1989). Il est à noter que le calcul de pose à partir de correspondances 3D-2D est une méthode rapide et contrairement à l'approche 2D-2D, il n'y a pas d'ambiguïté sur l'échelle.

1.3.4.1 *The Direct Linear Transformation*

La méthode *Direct Linear Transformation* (DLT) a d'abord été développée par les photogrammètres (Sutherland (1964)) puis introduite dans la communauté de vision par ordinateur (Faugeras (1993); Hartley et Zisserman (2000)). Elle peut être utilisée pour estimer tous les éléments de la matrice P de l'équation (1.4) en résolvant un système linéaire, même lorsque les paramètres internes ne sont pas connus. Chaque correspondance $\mathbf{Q}_i \leftrightarrow \mathbf{q}_i$ donne lieu à deux équations linéairement indé-

pendantes dans les entrées p_{ij} de P, qui sont

$$x_i = \frac{p_{11}X_i + p_{12}Y_i + p_{13}Z_i + p_{14}}{p_{31}X_i + p_{32}Y_i + p_{33}Z_i + p_{34}}, \tag{1.45}$$

$$y_i = \frac{p_{21}X_i + p_{22}Y_i + p_{23}Z_i + p_{24}}{p_{31}X_i + p_{32}Y_i + p_{33}Z_i + p_{34}}. \tag{1.46}$$

Ces équations peuvent être réécrites sous la forme $\mathsf{A}\mathbf{p} = 0$, où $\mathbf{p}$ est un vecteur composé des coefficients p_{ij}. La solution est le vecteur propre correspondant à la valeur propre minimale qui peut être trouvée à partir de la décomposition en valeurs singulières (SVD) de A.

Les paramètres internes et externes peuvent ensuite être extraits à partir de P (Faugeras (1993); Hartley et Zisserman (2000)). Dans la pratique, estimer les paramètres internes et externes simultanément dépend fortement de la géométrie et du nombre de correspondances. Dans des configurations favorables, environ vingt correspondances peuvent suffire alors que, dans des cas défavorables, même des centaines de correspondances peuvent ne pas être suffisantes. Dans de telles situations, il est toujours préférable d'estimer les paramètres internes séparément. Plus précisément, pour une application de localisation 3D, le fait d'utiliser une caméra calibrée et de n'estimer que son orientation et sa position $(\mathsf{R} \mid \mathbf{t})$, permet d'obtenir des résultats plus fiables.

Lorsque la matrice de calibrage K est connue, les paramètres extrinsèques peuvent être extraits de P à un facteur d'échelle près tel que $(\mathsf{R} \mid \mathbf{t}) \sim \mathsf{K}^{-1}\mathsf{P}$. La matrice 3×3 formée par les trois premières colonnes de la matrice $(\mathsf{R} \mid \mathbf{t})$ ne forme pas nécessairement une matrice de rotation, mais pour en obtenir une, une simple correction peut être réalisée (Zhang (2000)).

En conditions réelles, les coordonnées des pixels $\mathbf{q}_i$ sont généralement bruitées. Parce que l'erreur réelle minimisée par l'algorithme DLT est une erreur algébrique, les paramètres estimés de la caméra par cette méthode doivent être affinés par un processus itératif d'optimisation de l'erreur de reprojection non-linéaire (section 1.2.1).

1.3.4.2 *The Perspective-n-Point Problem*

La méthode DLT vise à estimer l'ensemble des onze paramètres de la matrice de projection. Elle pourrait également être appliquée à l'estimation de la pose, mais si les paramètres internes sont connus, on a alors une surparamétrisation. Les résultats sont alors moins stables et requièrent plus de mises en correspondances de points que nécessaire. Lorsque les paramètres internes sont estimés séparément, une approche plus satisfaisante consiste à utiliser explicitement cette information.

Cette approche connue sous le nom de *perspective-3-point problem* [1] (P3P) a quatre solutions dans le cas général. Pour plus de quatre mises en correspondance de points, la solution est en général unique, que les points 3D soient coplanaires ou non, tant qu'ils ne sont pas colinéaires.

Différentes approches P3P ont été proposées dans la communauté de vision par ordinateur (Fischler et Bolles (1981); Haralick *et al.* (1991); Quan et Lan (1999)). Nous présentons ici la méthode de Grünert (1841) pour laquelle chaque paire (i, j) de correspondances 3D-2D $\mathbf{Q}_i \leftrightarrow \mathbf{q}_i$ et $\mathbf{Q}_j \leftrightarrow \mathbf{q}_j$ fournit une contrainte sur les distances entre les points 3D et le centre optique $\mathbf{t}$ de la caméra. Ces distances, notées $x_i = \|\mathbf{Q}_i - \mathbf{t}\|$ et $x_j = \|\mathbf{Q}_j - \mathbf{t}\|$, doivent être estimées dans un premier temps, par le théorème d'Al-Kashi. Chaque paire fournit la contrainte suivante :

$$d_{ij}^2 = x_i^2 + x_j^2 - 2x_i x_j \cos\theta_{ij}, \tag{1.47}$$

où $d_{ij} = \|\mathbf{Q}_i - \mathbf{Q}_j\|$ est la distance connue entre les points 3D $\mathbf{Q}_i$ et $\mathbf{Q}_j$, et θ_{ij} est l'angle entre les deux rayons optiques partant du centre optique $\mathbf{t}$ et passant par les points $\mathbf{q}_i$ et $\mathbf{q}_j$. Cet angle peut être calculé par le produit scalaire des vecteurs directeurs normés $\mathbf{d}_i$ et $\mathbf{d}_j$ tel que $\cos\theta_{ij} = \mathbf{d}_i{}^\mathsf{T}\mathbf{d}_j$. La contrainte peut s'écrire de la façon suivante :

$$f_{ij}(x_i, x_j) = x_i^2 + x_j^2 - 2x_i x_j \cos\theta_{ij} - d_{ij}^2 = 0. \tag{1.48}$$

Dans le cas de trois points on obtient le système d'équations quadratiques suivant :

$$\begin{cases} f_{12}(x_1, x_2) &= 0 \\ f_{13}(x_1, x_3) &= 0 \\ f_{23}(x_2, x_3) &= 0 \end{cases} \tag{1.49}$$

où x_1, x_2 et x_3 sont trois inconnues. Ensuite, les méthodes de résolution sont différentes mais très proches les unes des autres. Celle de Grünert (1841) utilise les changements de variables pour réduire le nombre d'inconnues du système et ainsi obtenir un polynôme de degré 4. Les quatre solutions du polynôme sont alors calculées de manière explicite pour obtenir au final quatre triplets possibles de distances (x_1, x_2, x_3).

La méthode de trois points possède donc des solutions multiples. Pour lever l'ambiguïté et obtenir une solution unique, il faut ajouter une correspondance supplémentaire. Une approche simple consiste à résoudre le polynôme de degré 4 pour des sous-ensembles de trois points parmi les quatre points, et de retenir la solution commune. Enfin, pour extraire les informations de pose de la caméra à partir de ces distances, différentes solutions techniques peuvent être utilisées. Elles ont été comparées par Haralick *et al.* (1994).

Dementhon et Davis (1995) ont proposé une autre méthode nommée POSIT, pour résoudre le problème de l'estimation d'une pose pour $n \geq 4$. Il s'agit d'abord

1. Ce problème d'estimation de pose nécessite trois mises en correspondance 3D-2D de points.

de calculer une solution approchée en supposant que le modèle de projection de la caméra est orthographique et qu'il est à l'échelle, ce qui signifie qu'une estimation initiale de la position et de l'orientation de la caméra peut être obtenue par la résolution d'un système linéaire. Un poids est attribué à chaque point en se basant sur la pose estimée et il est appliqué aux coordonnées du point. Une nouvelle estimation de la projection est réalisée à partir des coordonnées mises à l'échelle. Ce processus est répété jusqu'à convergence. Cette méthode est simple à mettre en œuvre, mais elle est relativement sensible au bruit. En plus, elle ne peut pas être appliquée lorsque les points sont coplanaires. Oberkampf *et al.* (1993) proposent une approche similaire pour le cas planaire, mais les deux cas, coplanaire et non-coplanaire, doivent être explicitement distingués. Plus récemment, Lepetit *et al.* (2009) ont présenté une méthode plus rapide et plus précise.

1.3.5 Ajustement de faisceaux

Nous venons de présenter différentes méthodes permettant, une fois combinées, d'estimer la trajectoire d'une camera et de reconstruire la structure 3D de la scène observée sous forme d'un nuage de points, et ce à partir d'une séquence d'images. Ceci constitue l'estimation initiale de la géométrie pour l'ensemble ou pour une partie de la séquence vidéo. Une fois que l'on dispose de cette estimation initiale, il est recommandé de raffiner les paramètres de ce modèle (les positions des points et les poses de la caméra) à travers une étape d'optimisation. Une solution optimale peut être calculée par ajustement de faisceaux. C'est une technique mise au point par les photogrammètres et qui s'est largement répandue dans le domaine de la vision par ordinateur. De nombreux travaux sur le *Structure from Motion* (SfM) l'utilisent pour l'ultime étape des algorithmes de reconstruction. L'objectif est de réestimer simultanément les coordonnées des points 3D et les poses des caméras afin de minimiser la somme des carrés des écarts entre les points détectés dans les images et les reprojections obtenues à partir du modèle calculé. Un état de l'art des algorithmes d'ajustement de faisceaux a été publié par Triggs *et al.* (2000).

1.3.5.1 Erreur de reprojection

Lorsqu'un ensemble de points 3D et de caméras sont reconstruits à l'aide des méthodes définies précédemment, il est nécessaire de définir une erreur permettant de mesurer la qualité de cette reconstruction. L'idée générale de cette erreur est de mesurer la distance entre l'endroit où le point est détecté dans l'image et sa position estimée. Si des erreurs 3D ont été proposées (par exemple mesurer la distance entre le rayon optique issu de l'observation et le point 3D), il a été montré qu'il est généralement préférable d'utiliser une erreur 2D Lu *et al.* (2000), en particulier pour éviter que les points 3D au loin, aient une erreur plus importante que ceux près de la caméra du fait de leurs profondeurs.

La solution couramment retenue est l'erreur de reprojection (Mikhail *et al.* (2001)) qui est exprimée en pixels. Une représentation de cette erreur est donnée en figure 1.7. Elle consiste à mesurer la distance 2D entre l'observation du point 3D dans l'image (c'est à dire la position 2D du point d'intérêt) et la projection du point 3D reconstruit dans cette même image :

$$\mathbf{r} = \left\| \mathbf{q} - \pi\left(\tilde{\mathbf{P}}\tilde{\mathbf{Q}}\right) \right\|. \tag{1.50}$$

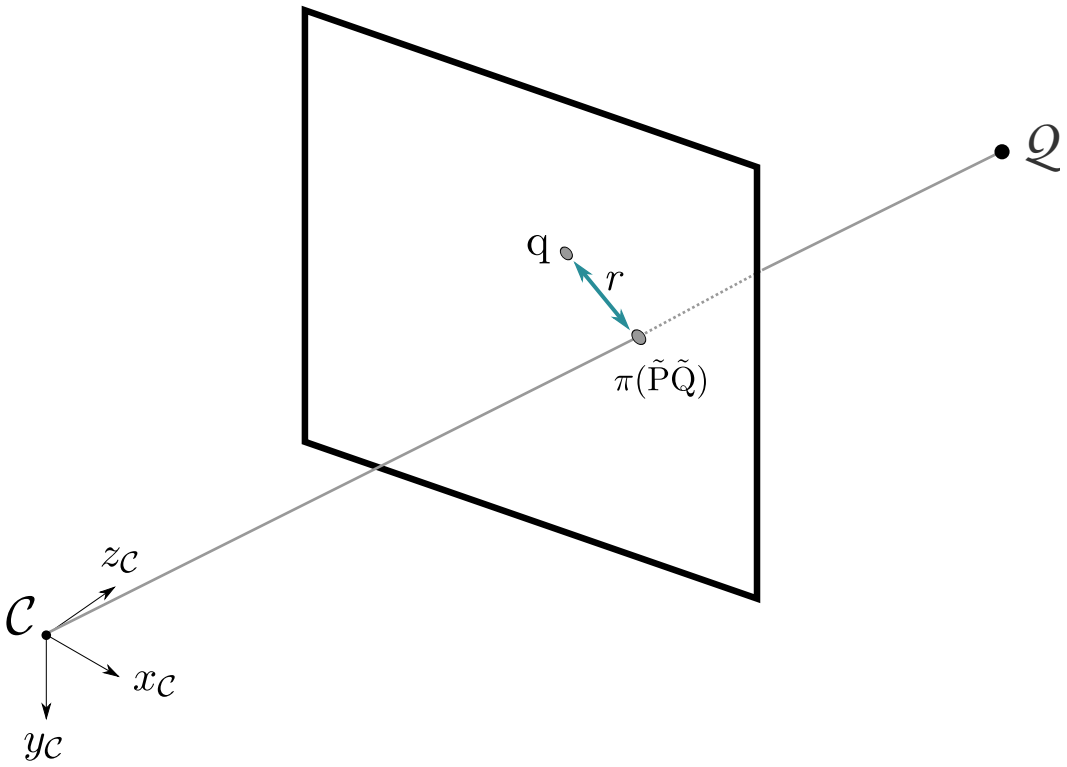


FIGURE 1.7 – **Erreur de reprojection d'un point.** L'erreur de reprojection d'un point 3D $\mathbf{Q}$ est la distance entre son observation 2D $\mathbf{q}$ et sa projection dans l'image $\pi\left(\tilde{\mathbf{P}}\tilde{\mathbf{Q}}\right)$.

1.3.5.2 Formulation du problème

Les méthodes de calcul de pose des caméras et de la structure de l'environnement, telles qu'elles ont été présentées précédemment, ne fournissent pas une solution optimale au problème de reconstruction et localisation simultanées. L'ajustement de faisceaux permet de raffiner l'ensemble des paramètres de la scène (à savoir, les 6 paramètres de pose de chaque caméra et les 3 paramètres de la position de chaque point 3D) en cherchant à minimiser l'erreur de reprojection pour chacun des couples caméra-point 3D observés.

Considérons un ensemble de points et de caméras dont les paramètres 3D ont été initialement estimés, l'ajustement de faisceaux consiste à améliorer ces paramètres afin d'augmenter la cohérence entre la reconstruction 3D et les données mesurées dans les images. Il s'agit donc de trouver les positions des points 3D et les poses des caméras (position + orientation) qui minimisent l'erreur de reprojection dans les images. Notons $\mathbf{x}$ le vecteur contenant l'ensemble des paramètres des points et des poses de la caméra à re-estimer. Il est de taille $N = 3 \times N_p + 6 \times N_c$ où N_p et N_c sont respectivement le nombre de points et de poses de caméra impliqués dans l'ajustement de faisceaux. Chaque point a trois paramètres (coordonnées 3D) et chaque pose de caméra a trois paramètres qui correspondent à la position 3D $\mathbf{t} = (t_x, t_y, t_z)$ du centre optique et trois paramètres (α, β, γ) qui correspondent à son orientation. Notons $\mathbf{y}$ le vecteur de mesure de taille M. Nous cherchons donc le vecteur de paramètres $\hat{\mathbf{x}}$ pour lequel l'erreur $\|\mathbf{r}\| = \|f(\hat{\mathbf{x}}) - \mathbf{y}\|$ est minimale. La fonction de coût de l'ajustement de faisceaux est donnée par l'équation suivante :

$$\mathcal{E}_E\left(\{\mathsf{R}_j, \mathbf{t}_j\}_{j=1}^{N_c}, \{\mathbf{Q}_i\}_{i=1}^{N_p}\right) = \sum_{i=1}^{N_p} \sum_{j \in \mathcal{A}_i} d^2(\mathbf{q}_{i,j}, \mathsf{P}_j \mathbf{Q}_i). \tag{1.51}$$

La reconstruction est ici composée de N_p points 3D $\{\mathbf{Q}_i\}_{i=1}^{N_p}$ et de N_c caméras $\{\mathsf{C}_k\}_{k=1}^{N_c}$. Notons $\mathbf{q}_{i,k}$ l'observation du point 3D $\mathbf{Q}_i$ dans la caméra $\mathcal{C}_k$ et $\mathcal{A}_i$ l'ensemble des indices des caméras observant $\mathbf{Q}_i$. La matrice de projection P_k associée à la caméra $\mathcal{C}_k$ est donnée par $\mathsf{P}_k = \mathsf{K}\mathsf{R}_k^\mathsf{T}\left(\mathcal{I}_3 | - \mathbf{t}_k\right)$, où K est la matrice des paramètres intrinsèques et $(\mathsf{R}_k, \mathbf{t}_k)$ les paramètres extrinsèques. $d^2(\mathbf{q}, \mathbf{q}') = \|\mathbf{q} - \mathbf{q}'\|^2$ est la distance point à point.

L'ajustement de faisceaux avec l'erreur de reprojection est un problème de minimisation non linéaire au sens des moindres carrés. Par conséquent, les techniques de minimisation présentées dans la section 1.2.1.2 peuvent être employées pour résoudre ce problème.

1.3.5.3 Résolution du problème

L'ajustement de faisceaux est une étape fondamentale en reconstruction 3D. Cependant, son implémentation est souvent considérée comme délicate car elle nécessite :

- ▷ le calcul des dérivées partielles,
- ▷ l'exploitation de la structure creuse de la matrice Jacobienne J.

Construction des matrices Jacobienne et Hessienne. La matrice Jacobienne $\mathsf{J} = [\mathsf{J}_c, \mathsf{J}_p]$ des résidus $\mathbf{r}$ en $\mathbf{x}$ représente les relations entre les paramètres de la reconstruction 3D (les poses de la caméra et les points 3D de la scène [2]) et les observations mesurées des points 2D. Rappelons que $\mathbf{x}$ est le vecteur des paramètres

2. Les indices $_c$ et $_p$ représentent respectivement la dépendances aux paramètres des poses de la caméra et des points 3D.

à estimer et qu'il contient les paramètres extrinsèques des caméras et les coordonnées des points 3D et que $\mathbf{r}$ est un vecteur concaténant toutes les erreurs résiduelles. L'étape principale d'une itération d'ajustement de faisceaux est l'estimation de l'incrément des paramètres $\delta_{LM} = [\delta_c, \delta_p]$ par résolution du système linéaire défini par l'équation :

$$\mathsf{N}'\delta_{LM} = \mathsf{J}^\mathsf{T}\mathbf{r}, \tag{1.52}$$

où la matrice N' est égale à la matrice $\mathsf{J}^\mathsf{T}\mathsf{J}$ (l'approximation de Gauss-Newton de la matrice Hessienne) dont les coefficients diagonaux sont multipliés par une constante dans le cas de l'algorithme Levenberg-Marquart.

Dans l'ajustement de faisceaux, les dérivées de la fonction de coût sont calculées analytiquement (le détail des équations des dérivées partielles est donné dans la thèse de Royer (2006)). En effet, l'utilisation de la matrice Jacobienne J implique le calcul des dérivées partielles. Cependant, tous les points ne sont pas observés dans toutes les images et la matrice J contient des coefficients nuls. Cela implique, une structure spéciale par blocs pour la matrice $\mathsf{J}^\mathsf{T}\mathsf{J}$ (voir la figure 1.8). Cette matrice est composée de trois matrices : U, V, et W telles que U et V sont diagonales par blocs Hartley et Zisserman (2000). La matrice N' peut être décomposée de la manière suivante :

$$\mathsf{N}' = \begin{pmatrix} \mathsf{U} & \mathsf{W} \\ \mathsf{W}^\mathsf{T} & \mathsf{V} \end{pmatrix}, \tag{1.53}$$

où :

- ▷ $\mathsf{U} = \mathsf{J}_c{}^\mathsf{T}\mathsf{J}_c$, matrice carrée (de taille $6N_c \times 6N_c$) contenant des blocs diagonaux 6×6 représentant les relations entre les mesures dans une image i et les paramètres de caméra associés ;
- ▷ $\mathsf{V} = \mathsf{J}_p{}^\mathsf{T}\mathsf{J}_p$, matrice carrée (de taille $3N_p \times 3N_p$) diagonale par blocs 3×3 et donc facilement inversible. Chaque bloc représente les relations entre les paramètres d'un point j et les observations qu'on a de ce point ;
- ▷ $\mathsf{W} = \mathsf{J}_c{}^\mathsf{T}\mathsf{J}_p$, matrice (de taille $6N_c \times 3N_p$) exprimant les intercorrélations entre les paramètres des points 3D et les paramètres des caméras. La structure de W est directement liée au fait que les points sont vus ou non dans les images. W a un nombre de blocs 6×3 non nuls égal au nombre de reprojections 2D.

Résolution éparse des équations normales par le complément de Schur. Pour comprendre la résolution éparse des équations normales, il faut expliciter la structure des matrices Jacobienne et Hessienne de la fonction de coût. Si l'on organise le vecteur de paramètres avec les paramètres des caméras puis des points, et le vecteur de résidus en regroupant les résidus liés à chaque image, la matrice Jacobienne J a la forme dessinée sur la figure 1.8. L'approximation de Gauss-Newton de la matrice Hessienne, dont la structure est montrée sur la figure 1.8, s'écrit $\mathsf{J}^\mathsf{T}\mathsf{J}$. Cette

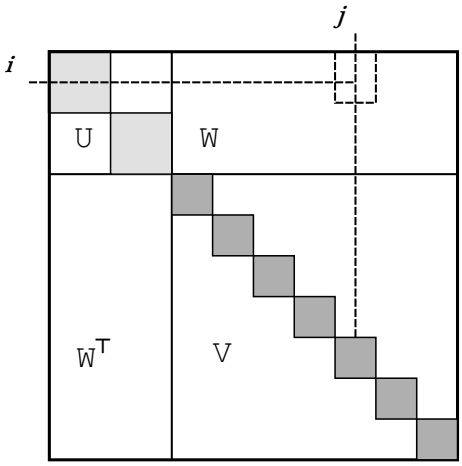


FIGURE 1.8 – Structure de la matrice Hessienne approchée $J^{\mathsf{T}}J$.

structure particulière de l'approximation de la matrice Hessienne rend possible la résolution de l'équation (1.52) de manière très rapide par le complément de Schur (Triggs *et al.* (2000)). Le système se décompose alors de la manière suivante :

$$\begin{pmatrix} \mathsf{U} & \mathsf{W} \\ \mathsf{W}^{\mathsf{T}} & \mathsf{V} \end{pmatrix} \begin{pmatrix} \delta_c \\ \delta_p \end{pmatrix} = \begin{pmatrix} \mathbf{g}_c \\ \mathbf{g}_p \end{pmatrix}, \tag{1.54}$$

avec $\mathbf{g}_c = \mathsf{J}_c{}^{\mathsf{T}}\mathbf{r}$ et $\mathbf{g}_p = \mathsf{J}_p{}^{\mathsf{T}}\mathbf{r}$. A l'aide du complément de Schur, ce système peut être résolu en deux étapes Hartley et Zisserman (2000) :

1. Le calcul du pas δ_c qui doit être appliqué aux paramètres des poses de la caméra par résolution du système linéaire suivant :

$$(\mathsf{U} - \mathsf{W}\mathsf{V}^{-1}\mathsf{W}^{\mathsf{T}})\delta_c = \mathbf{g}_c - \mathsf{W}\mathsf{V}^{-1}\mathbf{g}_p. \tag{1.55}$$

2. Le calcul direct du pas δ_p applicable aux points 3D :

$$\delta_p = \mathsf{V}^{-1}(\mathbf{g}_p - \mathsf{W}^{\mathsf{T}}\delta_c). \tag{1.56}$$

L'ajustement de faisceaux utilisant le complément de Schur est donc une technique très rapide et efficace puisque seule la matrice V, qui est diagonale par bloc 3×3, nécessite d'être inversée. Les équations normales résolues à chaque itération ont donc une structure éparse par blocs (Lourakis et Argyros (2009)). La prise en compte explicite de cette structure permet de résoudre ces équations avec une complexité réduite. Cette solution exploite la structure éparse correspondant aux paramètres des points. La complexité devient linéaire pour les points, mais reste cubique pour les poses de la caméra, c'est à dire qu'elle est réduite de $O(N^3M^3)$ à $O(NM^3)$.

1.4 Algorithme de localisation et reconstruction 3D utilisé

Dans cette section nous présentons l'algorithme de localisation et de reconstruction simultanées par vision monoculaire que nous utilisons. Il s'agit d'une méthode de type *Keyframe-based* SLAM qui permet de localiser une caméra en temps réel à partir d'une séquence vidéo. Ses différentes étapes sont décrites ci-dessous.

1.4.1 Présentation

La méthode de SLAM monoculaire que nous présentons dans ce mémoire est celle proposée par Mouragnon *et al.* (2006) et dont les grandes étapes sont résumées par l'algorithme 1. Comme nous le préciserons par la suite dans la section 2.1.2 du chapitre suivant, les algorithmes de SLAM monoculaire ont pour but de localiser une caméra dans un environnement inconnu. La résolution de ce problème s'effectue en passant par la construction en ligne d'une carte 3D de l'environnement à partir des observations extraites des images de la caméra au cours du temps. Ainsi, à chaque instant, la caméra observe des primitives déjà présentes dans la carte 3D reconstruite. Ces observations permettent alors de localiser la caméra. Lorsque cela est nécessaire, la carte est enrichie par l'ajout de nouvelles primitives ou par le raffinement de primitives déjà existantes. Ceci permet de reconstruire des zones de l'environnement qui n'ont pas encore été explorées.

Le principe général de cette méthode est schématisé sur la figure 1.9. Il s'agit d'une part de localiser la caméra au cours du temps, et d'autre part de reconstruire la scène de manière incrémentale en s'appuyant sur les reconstructions précédentes. Ce processus est possible grâce à un traitement 2D consistant à suivre (détecter, décrire et apparier) des points d'intérêt dans les images. Un algorithme d'ajustement de faisceaux local est régulièrement appliqué afin de réduire les erreurs accumulées.

Sans entrer dans les détails d'implémentation, nous allons présenter ci-après quelques unes des étapes spécifiques de la méthode, introduites dans Mouragnon *et al.* (2006). Cela permettra en particulier d'introduire des notions nécessaires à la compréhension de la suite du mémoire. Il est important de noter que notre approche est valable quelque soit la méthode de SLAM basé images clés utilisée.

1.4.2 Sélection des images clés

Dans la méthode de SLAM utilisée, toutes les images de la vidéo n'ont pas le même rôle. Certaines images seront utilisées uniquement pour localiser la caméra au moyen des primitives précédemment reconstruites. Les autres images, appelées images clés, ont un rôle particulier notamment pour l'étape de reconstruction 3D. Le déplacement d'une caméra vidéo entre deux vues consécutives étant généralement trop limité pour pouvoir appliquer les algorithmes de reconstruction avec préci-

Algorithme 1: Algorithme de SLAM monoculaire basé images clés.

Entrées : Une nouvelle image
// Traitements 2D
Détecter les $N = 300$ meilleurs points 2D [section 1.3.1];
Calculer les descripteurs de ces points d'intérêt [section 1.3.1];
Apparier ces points avec les 2 dernières images clés [section 1.3.1];
// Traitements 3D
Détecter si l'image courante est une image clé [section 1.4.2];
si *c'est une image clé* **alors**
  si *la scène n'est pas initialisée* **alors** // Processus d'initialisation
    si *il s'agit de la 3ème image clé détectée* **alors**
      // Calcul des poses initiales
      Estimer les 3 poses initiales à partir des correspondances 2D-2D [section 1.3.2];
      // Triangulation (Reconstruction 3D)
      Trianguler les points 3D [section 1.3.3];
      // Optimisation non linéaire
      Ajustement de faisceaux global [section 1.3.5];
    sinon
      Conserver la nouvelle image clé ;
  sinon // Processus incrémental
    // Calcul de pose 3D-2D
    Calcul des poses relatives entre les trois caméras clés à partir des mises en correspondances 2D-2D [section 1.3.4];
    // Triangulation (Reconstruction 3D)
    Trianguler les nouveaux points 3D [section 1.3.3];
    // Optimisation non linéaire
    Ajustement de faisceaux local sur les $N = 3$ dernières caméras [section 1.3.5];

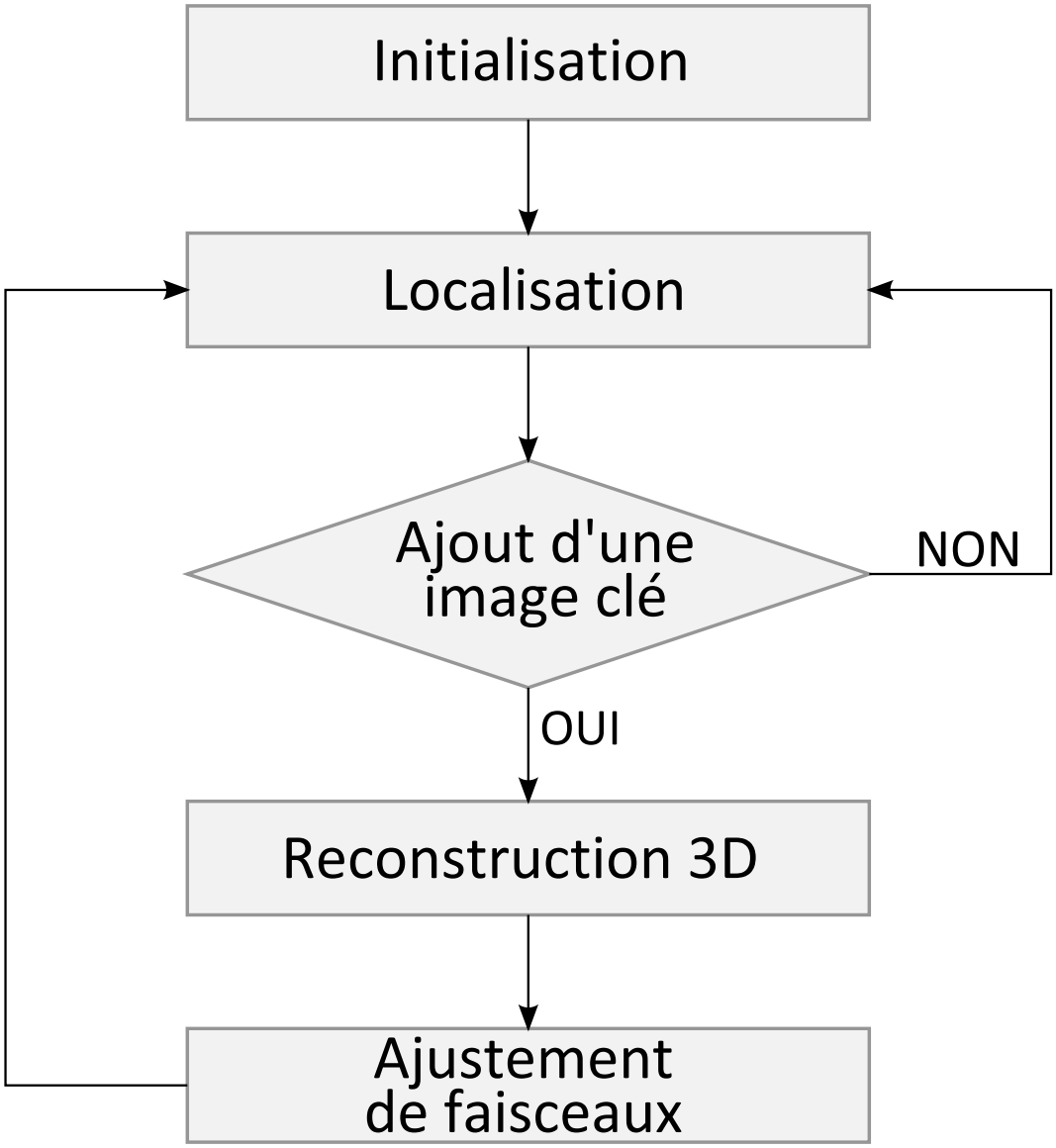


FIGURE 1.9 – Diagramme résumant le fonctionnement d'une méthode de type *keyfame-based* SLAM Mouragnon *et al.* (2006).

sion, le SLAM monoculaire de Mouragnon *et al.* (2006) propose de ne reconstruire la scène qu'avec certaines images, dites images clés. Pour cela, les auteurs définissent un critère permettant de définir si une image est une image clé ou non. Ce critère, essentiellement 2D, est notamment basé sur le nombre d'appariements 2D avec l'image clé précédente. Lorsque le nombre d'appariements 2D entre l'image courante et la dernière image clé est inférieur à un seuil, l'image précédente [3] est considérée comme une image clé.

D'autres critères pour la sélection d'images clés peuvent être envisagés. Par exemple Klein et Murray (2007) proposent de choisir les images clés selon un critère de proximité spatiale entre la pose de la caméra courante et les poses associées aux images clés précédentes.

1.4.3 Reconstruction et localisation initiales

Au début de la séquence d'images, l'algorithme de SLAM n'a pas encore d'informations sur la localisation de la caméra ni sur la structure de la scène. Mais avec l'ensemble des points d'intérêt détectés et appariés dans les trois premières images clés, il est possible de déterminer de manière robuste la trajectoire de la caméra et la structure initiale de la scène. Pour cela, la première pose $\mathcal{C}_1$ est fixée à l'origine c'est-à-dire à la position $(0, 0, 0)^\mathsf{T}$ et à l'orientation $\mathcal{I}_3$. Ensuite, le déplacement entre les images clés (associées aux poses $\mathcal{C}_1$ et $\mathcal{C}_2$ de la caméra) est calculé grâce aux correspondances 2D-2D entre ces deux images et à un calcul de la matrice Essentielle (voir la section 1.3.2.3). Les points 3D vus dans ces deux images clés sont triangulés (voir la section 1.3.3) et la pose associée à la troisième image clé (associée à la pose $\mathcal{C}_3$ de la caméra) est calculée par mise en correspondance 3D-2D (voir la section 1.3.4.2).

Cette première reconstruction 3D (les trois poses de la caméra et la scène) est ensuite raffinée au moyen d'un ajustement de faisceaux global (voir la section 1.3.5). Une fois la reconstruction initialisée et raffinée, le processus incrémental suivant est lancé.

1.4.4 Reconstruction et localisation incrémentales

Dès lors que l'initialisation est réalisée, pour chaque image clé détectée, la pose de la caméra est calculée grâce aux points 3D préalablement reconstruits. Les appariements 2D fournis par le module de suivi permettent d'obtenir des correspondances 3D-2D entre les points d'intérêt de l'image courante et les points 3D déjà reconstruits. La pose de la caméra courante est alors estimée à partir de ces correspondances (voir la section 1.3.4.2). De nouveaux points 3D qui n'ont pas encore été reconstruits et observés dans les trois dernières images clés, sont reconstruits par

3. C'est l'image précédente qui est sélectionnée car c'est la dernière image ayant eu un nombre de correspondances supérieur au seuil.

triangulation (voir la section 1.3.3) à l'aide du nouveau déplacement inter-caméras clés calculé précédemment. Un raffinement local est ensuite périodiquement appliqué, à chaque image clé, sur les dernières poses de la caméra et sur la scène pour réduire les erreurs accumulées.

1.4.5 Ajustement de faisceaux local

La particularité des travaux de Mouragnon *et al.* (2006) est que l'ajustement de faisceaux ne raffine pas l'ensemble de la reconstruction, on parle alors d'ajustement de faisceaux local (ou glissant). Le principe consiste à n'optimiser que les derniers paramètres. En effet, la complexité de l'ajustement de faisceaux par la méthode du complément de Schur est cubique par rapport au nombre de ses paramètres. Si l'on souhaite utiliser l'ajustement de faisceaux dans une application temps-réel, il est nécessaire de limiter les paramètres à optimiser. L'idée est de réduire le nombre des paramètres à estimer en optimisant uniquement un sous ensemble des points 3D reconstruits et des poses de la caméra. Dans son implémentation originale, les paramètres à optimiser sont les N plus récentes poses de la caméra (où N est choisi de manière à maintenir des performances de traitement temps réel) et les points 3D qu'elles observent. Dans Klein et Murray (2007), les caméras et les points 3D optimisés dans l'ajustement de faisceaux local sont sélectionnés avec un critère spatial [4]. Les auteurs de Mouragnon *et al.* (2006) ont montré que l'optimisation locale des $N = 3$ dernières poses de la caméra, était suffisante pour localiser et cartographier de manière relativement précise, un environnement inconnu. Une illustration de l'ajustement de faisceaux local est proposée en figure 1.10.

La sélection des points 3D à optimiser peut varier suivant les méthodes. Dans cette thèse, nous optimisons l'ensemble des points 3D vus dans les N dernières caméras. La fonction de coût de l'ajustement de faisceaux local comprend les erreurs de reprojection de ces points dans toutes les images où ceux-ci apparaissent. Notons que les points 3D vus dans moins de trois caméras ou ayant une erreur de reprojection supérieure à un seuil de quatre pixels (*outlier*) ne sont pas optimisés.

La résolution du problème de l'ajustement de faisceaux local est ensuite effectuée par le même procédé que pour l'ajustement de faisceaux global, décrit dans la section 1.3.5, mais en un temps beaucoup plus faible compte tenu du nombre restreint des paramètres simultanément optimisés.

4. Notons que dans Klein et Murray (2007) une implémentation parallèle du suivi et de la reconstruction est réalisée dans deux *threads* différents. De ce fait, un ajustement de faisceaux global peut également être réalisé, tant que les performances temps réel sont maintenues.

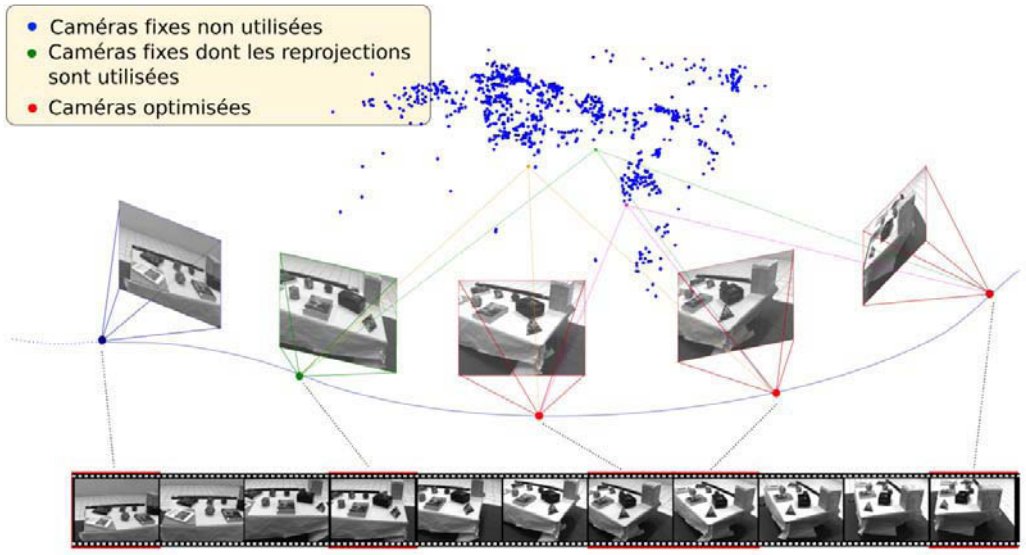


FIGURE 1.10 – **Ajustement de faisceaux local.** Les caméras et les points rouges sont optimisées, les erreurs de reprojection des caméras vertes sont intégrées dans la fonction de coût et les caméras bleues ne sont pas prises en compte.

CHAPITRE

2

Etat de l'art des méthodes de localisation par vision

Ce chapitre présente l'état de l'art des approches de localisation par vision. Il distingue les méthodes de localisation sans connaissance a priori sur l'environnement, les méthodes exploitant une connaissance sur cet environnement (issue d'un modèle 3D d'un objet de la scène) et enfin les méthodes considérant une caméra évoluant dans un environnement partiellement connu (c'est-à-dire un environnement inconnu avec, comme unique a priori un modèle 3D d'un objet de la scène).

2.1 Localisation en environnement inconnu

La localisation par vision monoculaire repose sur l'interprétation du mouvement 2D d'éléments d'intérêt dans les images successives. Le déplacement de la caméra dans l'espace 3D se traduit dans l'image par un déplacement en 2 dimensions de ces éléments. Ces derniers sont généralement des zones ou, plus couramment, des points qu'il est possible de suivre d'une image à l'autre. A partir de l'observation du déplacement de ces éléments d'intérêt, il est possible d'inférer le mouvement de la caméra. Ceci peut être réalisé directement à partir de l'information 2D en s'appuyant par exemple sur la géométrie épipolaire liant deux images (Tardif *et al.* (2008)). Une autre approche qui s'est fortement développée ces dernières années consiste à passer par l'intermédiaire de la création d'une carte 3D de l'environnement. Nous parlerons dans la suite de SfM (*Structure from Motion*) puis de SLAM (*Simultaneous Localization And Mapping*). Ces deux méthodes se distinguent par l'objectif visé : le SfM vise une reconstruction 3D de la scène avec la meilleure qualité possible alors que le SLAM se concentre sur une localisation temps réel. Notons que ces méthodes reposent sur l'hypothèse d'une scène rigide.

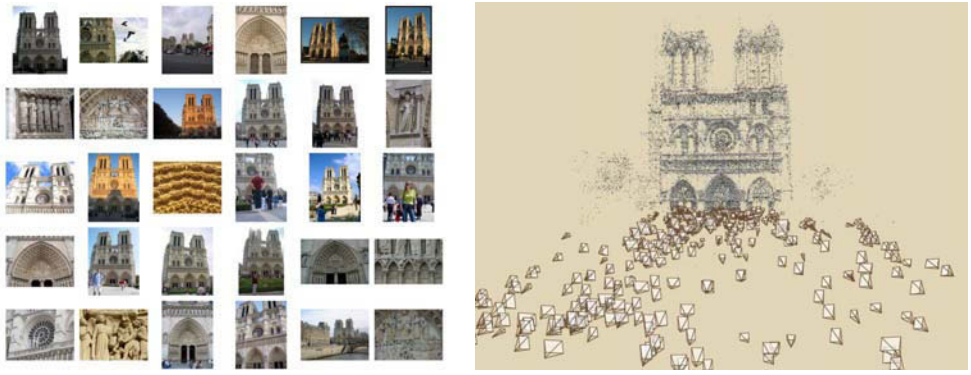
2.1.1 *Structure from Motion* (SfM)

Dans le domaine de la vision par ordinateur, les méthodes de type SfM consistent à inférer de manière automatique, à partir d'un ensemble d'images, la structure 3D de la scène observée et les positions des caméras à partir desquelles ces images ont été acquises. Une recherche très active a été menée sur ce thème durant ces trente dernières années (Ullman (1979); Bao et Savarese (2011)). Ces recherches ont abouti à un état de maturité tel que certaines d'entre elles ont permis la commercialisation de produits tels que `PhotoTourism 2011` ou encore `AutoStitch 2011` pour la création de panoramas. La figure 2.1 illustre un exemple de reconstruction 3D obtenue par la méthode décrite dans Snavely *et al.* (2006, 2008) aujourd'hui utilisée par `PhotoTourism`.

Les origines du SfM viennent de la photogrammétrie, qui depuis la deuxième moitié du XIX^{ème} siècle, se consacre à l'extraction d'informations géométriques d'une scène à partir d'images de celle-ci. Pour ce faire, l'utilisateur commence par une identification manuelle d'un ensemble de primitives 2D qui sont ensuite reconstruites en 3D par triangulation. Les méthodes de SfM utilisent, pour finir, des techniques d'optimisation non linéaire pour minimiser l'erreur de reprojection de ces primitives mesurées dans les images pour améliorer simultanément la structure de la scène et les poses de la caméra. Ces techniques sont connues sous le nom d'ajustement de faisceaux ou *Bundle Adjustment* (BA) en anglais (Triggs *et al.* (2000)). Les recherches en vision par ordinateur ont principalement été orientées pour obtenir une complète automatisation de ce système. Des progrès remarquables, sur trois aspects, ont été obtenus par cette communauté de recherche. Le premier étant la formalisation des contraintes géométriques imposées sur le mouvement par l'observation des primitives à partir de plusieurs vues, sous l'hypothèse de rigidité de la scène, dans le cas où la caméra n'est pas calibrée (Hartley et Zisserman (2000)). Le deuxième aspect d'amélioration concerne la détection et la description de primitives avec un fort degré de stabilité (tel que Canny (1986); Harris et Stephens (1988); Lowe (2004)). Le dernier aspect concerne le rejet de données aberrantes. Le résultat de ces avancées scientifiques a mené à l'élaboration de systèmes automatiques de mise en correspondance de primitives (points, droites, segments, *etc.*) dans des images et ainsi qu'à l'estimation de la transformation géométrique entre les poses des caméras associées à ces images.

Basées sur ces trois améliorations, plusieurs méthodes ont été proposées qui, à partir d'un ensemble d'images discrètes d'une scène, sont capables d'estimer la structure 3D et les différentes localisations de la caméra à une transformation projective près, dans le cas général de caméra non calibrée ou à facteur d'échelle près, dans le cas d'une caméra calibrée. Nous avons vu que cette estimation de SfM, à partir de deux ou trois vues, est habituellement (comme en photogrammétrie) suivie d'une étape d'ajustement de faisceaux qui améliore l'estimation initiale en optimisant la consistance globale. L'optimisation repose sur la minimisation d'une fonction de coût non linéaire. La fonction de coût de l'ajustement de faisceaux est

donnée par l'équation (1.51). Notons que la prise en compte du système creux de l'ajustement de faisceaux (Triggs *et al.* (2000)) permet d'augmenter de manière significative la rapidité de ces méthodes, et de traiter un très grand nombre d'images. Ces méthodes permettent alors d'obtenir des reconstructions 3D très précises sur de grandes scènes (Agarwal *et al.* (2009); Frahm *et al.* (2010)).



(a) (b)

FIGURE 2.1 – Le système de SfM proposé par Snavely *et al.* (2006) utilise un ensemble d'images (a) d'une scène (ici des photos obtenues par une recherche sur internet) pour reconstruire cette dernière (b) sous forme d'un nuage de points 3D et estimer les poses des caméras correspondant aux points de vue à partir desquels les photos ont été obtenues.

2.1.2 SLAM monoculaire

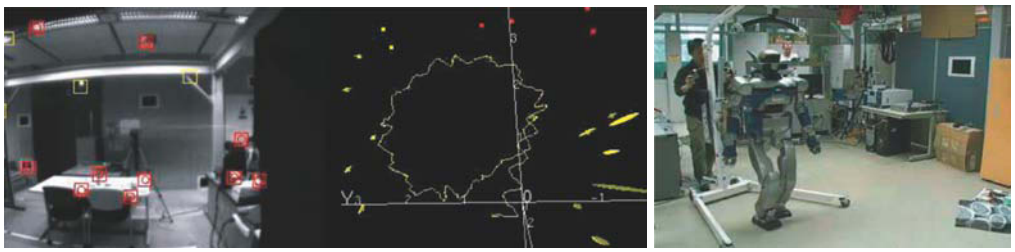
L'estimation du mouvement d'une plateforme mobile et de l'environnement statique dans lequel elle se déplace, a également été traitée par la communauté de robotique mais d'un point de vue légèrement différent. L'algorithme bien connu sous le nom de SLAM (pour *Simultaneaous Localization And Mapping*) a été considéré comme l'un des problèmes fondamentaux dans le domaine de la robotique mobile. Le SLAM adresse l'estimation du mouvement d'un robot mobile et de son environnement à partir de données fournies par un ou plusieurs capteurs. Les premières approches de type SLAM utilisent plusieurs capteurs, tels que le laser (Castellanos *et al.* (1999)), le radar (Dissanayake *et al.* (2001)), le sonar (Tardos *et al.* (2002)), ou encore l'odométrie fournie par le mouvement des roues. La fusion de données est généralement l'axe principal de ces recherches (Castellanos *et al.* (2001)). Comme, parallèlement à cela, les résultats en vision par ordinateur devenaient suffisamment matures, la vision a pris une part prépondérante dans ces schémas de fusion de données. Elle fût même utilisée comme l'unique capteur d'entrée pour le SLAM. On peut notamment citer les travaux de Davison *et al.* (2007) illustrés par la figure 2.2. Le SLAM monoculaire réfère donc à l'utilisation d'une seule caméra comme

principal ou unique capteur pour réaliser la localisation. La différence principale entre le *Structure from Motion* et le SLAM monoculaire est la suivante : alors que le SfM permet de gérer le problème de localisation et de reconstruction 3D dans le cadre général, c'est-à-dire quelque soit l'entrée visuelle, le SLAM monoculaire se concentre sur des approches séquentielles pour la gestion d'une vidéo comme donnée d'entrée. Ceci est lié à la spécificité de l'application robotique pour laquelle l'estimation en temps réel de la pose courante du robot mobile est essentielle. Cette contrainte de traitement temps réel n'est pas restreinte aux applications robotiques : par exemple, la réalité augmentée nécessite également une estimation temps réel de la pose courante de la caméra dans le but d'insérer, de manière fluide et cohérente, des éléments virtuels sur chaque image de la séquence vidéo. La figure 2.3 représente un résultat de localisation obtenu avec la méthode de type SLAM monoculaire proposée par Klein et Murray (2007) pour des applications de réalité augmentée.

Généralement, pour limiter la dérive des paramètres estimés, une estimation simultanée de la structure de la scène et des poses de la caméra doit être réalisée intégrant ainsi la contrainte de rigidité de la scène. Pour cela, deux types de méthodes ont été proposés :

- ▷ les méthodes *Keyframe-based* SLAM reposant sur l'ajustement de faisceaux local,
- ▷ les méthodes *Filtering-based* SLAM reposant sur les techniques d'estimation par filtrage.

Nous allons présenter dans la suite, ces deux principales familles d'approches de SLAM.



(a) (b)

FIGURE 2.2 – Illustration d'un résultat de localisation par la méthode de SLAM proposée dans Davison *et al.* (2007) et connue sous le nom de MonoSLAM. Ce résultat (a) décrit la trajectoire circulaire d'un rayon de 0,75 mètre, réalisée par un robot humanoïde (b). La trace jaune représente la trajectoire estimée du robot, et les ellipses représentent la position des primitives 3D reconstruites et leurs incertitudes.

(a) (b)

FIGURE 2.3 – Résultats de localisation obtenus avec la méthode de Klein et Murray (2007) et utilisés pour des applications de réalité augmentée.

2.1.2.1 *Keyframe-based* SLAM

Cette approche de SLAM, adapte les techniques de SfM pour un traitement séquentiel en temps réel (Nistér *et al.* (2004); Engels *et al.* (2006); Mouragnon *et al.* (2006)). Celle-ci consiste à utiliser en ligne et de façon incrémentale les outils classiques de vision par ordinateur (triangulation, calcul de pose, ajustement de faisceaux, *etc.*). La reconstruction 3D ainsi que la localisation du capteur se font dans ce cas au fur et à mesure que de nouvelles images arrivent, ce qui en fait une solution pour le SLAM monoculaire.

Le principe général de ces méthodes consiste à estimer la pose de la caméra à partir des primitives déjà connues dans la carte 3D, à reconstruire de nouvelles primitives 3D à chaque fois que cela est nécessaire et enfin à raffiner de manière conjointe les primitives et les poses de la caméra pour un ensemble de caméras clés choisies dans la séquence, d'où le nom de *Keyframe-based*. Pour les méthodes de SfM (décrites en section 2.1.1), l'ajustement de faisceaux est réalisé hors ligne car cette tache est très coûteuse en temps de calcul. La complexité algorithmique de l'étape d'ajustement de faisceaux est en $O(N^3)$, où N est le nombre de primitives 3D reconstruites. La principale innovation des méthodes *Keyframe-based* SLAM repose sur l'adaptation de l'ajustement de faisceaux pour obtenir des performances temporelles compatibles avec le traitement temps réel d'une séquence vidéo. L'ajustement de faisceaux est réalisé uniquement sur un ensemble d'images clés judicieusement choisies. Ces images clés peuvent être choisies sur une fenêtre temporelle glissante autour de l'image courante (Mouragnon *et al.* (2006)) ou encore sur un critère spatial comme dans Klein et Murray (2007, 2008) [1]. La résolution du problème de l'ajustement de faisceaux local, à chaque nouvelle image clé, est ensuite

1. Ils proposent en plus de traiter sur deux *thread* différents la reconstruction et l'optimisation. Cela permet de réaliser, en parallèle de la localisation, un raffinement global de la carte 3D.

effectuée par le même procédé que pour l'ajustement de faisceaux global (décrit par l'équation (1.51)). Le fait de n'optimiser qu'un nombre restreint de paramètres permet de réaliser cette tache en temps réel.

Les nombreux progrès réalisés en vision font qu'à l'heure actuelle, ces méthodes permettent d'obtenir en temps réel une localisation de la caméra sur de longues distances (plusieurs kilomètres Civera *et al.* (2009); Konolige *et al.* (2007)).

2.1.2.2 *Filtering-based* SLAM

Le principe de ce type d'approche, consiste en la construction d'une carte probabiliste des primitives, représentant à chaque instant un aperçu des estimations courantes de l'état de la caméra et de toutes les primitives. Cette approche fournit aussi et surtout, l'incertitude de ces estimations. La carte est initialisée au démarrage du système et évolue continuellement et dynamiquement en étant mise à jour par des filtres bayésiens tels que le filtre de Kalman, le filtre de Kalman étendu ou encore le filtre à particules. Ce sont des outils probabilistes généralement utilisés pour l'estimation de l'état d'un système dynamique. Dans le cas du SLAM, les estimations de l'état de la caméra et des primitives 3D reconstruites sont mises à jour au cours du déplacement de la caméra et des observations des primitives. Lorsque de nouvelles primitives sont observées, la carte est agrandie et si cela est nécessaire, des primitives peuvent également être supprimées.

Le caractère probabiliste de la carte réside dans la propagation au fil du temps, non seulement des estimations des différents états de la caméra et des primitives, mais aussi d'une distribution au premier ordre de l'incertitude permettant de décrire les écarts possibles par rapport à ces estimations. Mathématiquement, la carte est représentée par un vecteur d'état et une matrice de covariance. Le vecteur d'état est composé de la concaténation des estimations de la caméra et des primitives ; la matrice de covariance est carrée et de dimension égale à celle du vecteur d'état. La distribution de probabilité de l'ensemble des paramètres de la carte est généralement approximée par une distribution gaussienne à variables multiples dans un espace de dimension égale à celui du vecteur d'état. La taille totale de la représentation de la carte est de l'ordre de $O(N^2)$, où N est le nombre de primitives. L'algorithme de SLAM complet a une complexité en $O(N^2)$. Cela signifie que le nombre de primitives qui peuvent être maintenues, avec un traitement en temps réel, est limité. Donc plutôt que de servir à une description complète de la scène, le rôle de la carte est principalement de permettre une localisation de la caméra en temps réel. Généralement la scène est représentée par un ensemble de primitives qui peuvent être des points (Davison *et al.* (2007)), des segments (Gee et Mayol-Cuevas (2006)), des plans (Servant *et al.* (2008) [2]), *etc.*

2. L'utilisation des plans permet tout d'abord d'améliorer l'estimation puisqu'ils fournissent plus d'informations que les points habituellement utilisés. De plus en permettant une factorisation de la carte par regroupement des éléments appartenant au même plan, les coûts calculatoires sont réduits.

Parfois, la caméra se déplace autour d'un espace 3D restreint. Cela implique que les primitives entrent et sortent du champ de vue de la caméra en fonction de son déplacement. Dans ce cas des fermetures de boucles peuvent être envisagées (le lecteur intéressé pourra notamment se référer aux travaux de Angeli *et al.* (2008); Ho et Newman (2007)). Il est alors fortement préférable de représenter, avec précision, et flexibilité, les corrélations qui surviennent entre les différentes parties de la carte. Pour cela, une mise à jour est généralement réalisée dans un unique vecteur d'état des primitives ainsi qu'une mise à jour de la matrice de covariance associée. Une centaine de primitives bien choisies avec une gestion appropriée de la carte (Dissanayake *et al.* (2002)) s'avère généralement suffisantes afin de couvrir l'ensemble d'une pièce. A noter que des approches basées sur un filtre à particule telles que Montemerlo *et al.* (2002); Eade et Drummond (2006b) usuellement appelées FastSLAM permettent de gérer un plus grand nombre de primitives, mais représentent les corrélations avec moins de précision ce qui ne garantit pas de fournir d'aussi bons résultats au niveau des relocalisations pour les fermetures de boucle.

Par ailleurs, on peut noter que certaines méthodes (Davison *et al.* (2007)) maintiennent également des estimations de la vitesse linéaire et angulaire. Cette information est utilisée dans l'algorithme comme dynamique du mouvement de la caméra.

2.1.2.3 *Keyframe-based* vs *Filtering-based* SLAM

Nous venons de voir, deux approches de type SLAM pour résoudre le problème de la localisation d'une caméra. Les méthodes de type *Filtering-based* SLAM et de *Keyframe-based* SLAM passent toutes les deux par l'intermédiaire de la création, en ligne, d'une carte de l'environnement, pour localiser en temps réel une caméra dans un milieu inconnu. D'un point de vue probabiliste, le SLAM peut être, plus généralement, représenté comme un réseau bayésien (Thrun *et al.* (2005)) représentant des variables aléatoires sous la forme d'un graphe orienté acyclique. Cela permet de modéliser l'estimation de la structure 3D et de la trajectoire de la caméra par un champ aléatoire de Markov [3] (voir la figure 2.4). Il faut tout de même prendre en compte le fait que le graphe évolue à chaque nouvelle pose de caméra (notée $X_{0..3}$ dans la figure 2.4), et à chaque fois que l'on ajoute de nouvelles primitives 3D (notées $y_{1..6}$ dans la figure 2.4). Les observations des primitives dans les images doivent également être ajoutées car elles lient les poses de la caméra avec les primitives 3D. L'objectif du SLAM monoculaire consiste à propager, à chaque image, les nouvelles observations des primitives 3D, à toutes les caméras les ayant déjà observées.

Sur ce schéma les approches de type *Filtering-based* SLAM ajoutent, à chaque étape de l'estimation, les paramètres des caméras clés $X_{0,3}$ et les paramètres des

3. Dans ces modèles, les relations d'interdépendances sont décrites par un graphe non orienté, en tenant compte de la caractéristique markovienne exprimant que la dépendance spatiale des primitives 3D et des poses de la caméra, ne provient que des voisins immédiats.

primitives 3D observées par ces caméras $y_{1..6}$. Rappelons que les approches de type *Filtering-based* SLAM utilisent des filtres bayésiens, tels que le filtre de Kalman ou le filtre à particules pour l'estimation de l'état de la caméra et de la carte 3D reconstruite. Dans ce cas, à chaque étape de l'estimation, seul les paramètres de la caméra courante X_3 sont ajoutés, sans prendre en compte les précédentes poses $X_{0..2}$. De plus une mise à jour complète des relations d'interdépendances du graphe est réalisée, à partir des mesures de l'image courante (représentées dans la figure 2.4 par des arcs de cercle entre les différentes primitives $y_{1..6}$). Contrairement à l'optimisation d'images clés par ajustement de faisceaux, le filtrage ne prend pas en compte les poses du passé et conserve une estimation probabiliste jointe de la pose courante et de la carte 3D des primitives. Notons cependant qu'il conserve et propage l'historique des poses précédentes. Cela est réalisé grâce à l'équation d'état et au calcul de la confiance à lui accorder par un gain automatique calculé à partir des covariances.

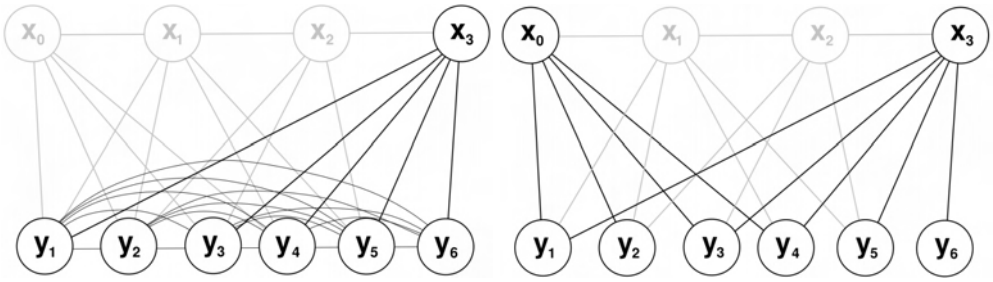
Il est tout à fait naturel de se demander quelle approche entre le *Keyframe-based* SLAM et le *Filtering-based* SLAM fournit de meilleurs résultats. Elles ont toutes les deux leurs avantages et inconvénients. Le *Keyframe-based* est réputé pour être plus précis et le *Filtering-based* SLAM plus adapté pour faire de la fusion de données ou pour traiter des données momentanément erronées. Il est en pratique très difficile de dire qu'une méthode est meilleure que l'autre car cela est très dépendant de l'application visée. Toutefois, certaines études notamment celle de Strasdat *et al.* (2010) ont mis en avant que l'approche de type *Keyframe* avec ajustement de faisceaux est meilleure que celle de type *Filtering* en termes de précision et de coût de calcul. Notons de plus que de récents travaux sur le *Keyframe-based* SLAM permettent de combler certaines lacunes qui existaient par rapport au *Filtering-based* SLAM. A savoir, la propagation des covariances à travers l'ajustement de faisceaux (Eudes et Lhuillier (2009)), la fusion de capteur dans l'ajustement de faisceaux (Eudes *et al.* (2010); Michot *et al.* (2010); Lhuillier (2012)) et l'exploitation de primitives déjà reconstruites pour gérer la tâche de fermeture de boucles (Klein et Murray (2007)).

2.1.3 Limites de ce type de méthodes

Si les méthodes de type SLAM actuelles permettent de reconstruire la trajectoire d'une caméra dans un environnement inconnu, elles présentent néanmoins encore aujourd'hui des limites qui empêchent leur utilisation pour un grand nombre d'applications.

2.1.3.1 Localisation non géoréférencée

Dans l'approche SLAM, aucune information *a priori* n'est disponible sur l'environnement dans lequel la caméra évolue. Le SLAM fournit le déplacement relatif de la caméra au fil du temps par rapport au repère initial de la caméra, généralement



(a) *Filtering-based* SLAM (b) *Keyframe-based* SLAM

FIGURE 2.4 – L'estimation par le SLAM d'une structure 3D et de la trajectoire d'une caméra peut être modélisé par un champ aléatoire de Markov (figure extraite de Strasdat *et al.* (2010)). Cette représentation permet de se faire une rapide idée des différences entre les approches de type *Filtering-based* (a) et *Keyframe-based* (b) SLAM. Les poses de la caméra sont notées $X_{0..3}$ et les primitives 3D sont notées $y_{1..6}$. En foncé, sont représentées les éléments pris en compte à chaque nouvelle image clé.

choisi arbitrairement. Donc la caméra n'est pas localisée par rapport à un repère absolu. On parle alors d'une localisation non géoréférencée.

2.1.3.2 Dérives sur les longues trajectoires

La cause principale de la dérive observée dans les processus de SLAM est l'erreur qui existe sur les mesures effectuées dans les images. En effet, l'ensemble du système SLAM s'appuie sur des primitives (points, segments,...) détectées dans les images successives. Cependant, l'étape de détection des primitives n'est pas parfaite. Les coordonnées de ces dernières présentent un bruit, généralement supposé gaussien. Naturellement, les erreurs effectuées sur les mesures 2D ont un impact direct sur la reconstruction 3D des primitives observées et sur la localisation de la caméra.

Accumulation d'erreur. Le processus de SLAM est un processus incrémental. La position reconstruite de la caméra à l'instant t dépend des données collectées jusqu'à l'instant $t-1$. Or, l'estimation de l'état courant (i.e. la position de la caméra et des primitives dans la carte) à partir des données de l'instant précédent est entachée d'erreurs. En effet, des erreurs directement liées à la méthode (erreurs dans les données observées) et à son exécution (précision de calcul limitée sur les machines) impliquent une erreur sur le calcul du déplacement relatif entre les instants $t-1$ et t. De plus, à l'erreur réalisée sur l'estimation du déplacement entre les instants $t-1$ et t s'ajoute l'erreur initialement réalisée sur l'état à l'instant $t-1$. Au cours du processus de SLAM, les erreurs s'ajoutent à chaque nouvelle estimation : on parle alors

du phénomène d'accumulation d'erreur. Dès lors, les méthodes de SLAM peuvent difficilement être utilisées en l'état pour des applications de recalage de modèle 3D.

Dérive du facteur d'échelle. Dans le cas du SLAM monoculaire, la reconstruction est réalisée à un facteur d'échelle près. Ce facteur est fixé arbitrairement au début du processus et doit être propagé tout au long de la séquence. En monoculaire, il est difficile de propager ce facteur car il n'est pas observable au cours de la séquence. Il subit directement l'accumulation des erreurs et dérive au cours du temps, en particulier lorsque le mouvement de la camera est tel que le suivi d'un grands nombre de primitives est perdu entre deux images (mouvement brusque de la caméra, virage, *etc.*).

2.2 Localisation par rapport à un modèle 3D connu

Nous présentons dans cette section les méthodes qui exploitent uniquement un modèle de l'environnement ou d'un objet de la scène, pour estimer le mouvement de la caméra (c'est-à-dire qu'à la différence du SLAM, elles ignorent le reste de la scène). Dans ce cas il n'y a plus de reconstruction 3D de primitives, seules celles extraites du modèle 3D sont utilisées pour la localisation. Le principe est donc le suivant :

1. établir une mise en correspondance entre les primitives de l'image courante et les primitives 3D du modèle,
2. déterminer les six degrés de liberté de la pose qui minimise la distance entre la projection des primitives 3D et de leurs correspondants 2D.

Les méthodes employées pour ces étapes dépendent principalement de la nature du modèle. En effet, pour modéliser un objet, plusieurs possibilités existent : on peut modéliser la forme de l'objet (modèle géométrique), l'apparence (modèle photométrique), ou les deux (modèle photo-géométrique). Nous aborderons ensuite les méthodes qui mettent à jour ce modèle au cours du suivi. Pour une comparaison plus complète de ce type de méthodes, le lecteur pourra se référer à la revue proposée par Lepetit et Fua (2005).

2.2.1 Méthodes basées sur un modèle 3D géométrique

Cette première famille de solution de localisation exploite un modèle purement géométrique de l'objet, c'est-à-dire caractérisant uniquement la forme 3D de l'objet. Le principe de ces méthodes est relativement simple puiqu'il s'agit de mettre en correspondance des primitives 3D issues du modèle avec les primitives 2D leur correspondant dans l'image courante (*eg.* points 3D du modèle avec des points d'intérêt de l'image, arête 3D du modèle avec segment de l'image, *etc.*). Néanmoins, cette

mise en correspondance est généralement difficile à établir étant donné le faible pouvoir discriminant de ces primitives.

Pour y parvenir, une solution consiste à établir simultanément l'ensemble des mises en correspondance, celles-ci étant déterminées de manière à fournir une pose minimisant l'erreur de reprojection des primitives 3D par rapport à leur primitive 2D appariée (David *et al.* (2004)). Cependant, pour permettre de réaliser ce traitement en temps réel, une hypothèse supplémentaire est généralement ajoutée : celle de faible mouvement. Cette hypothèse consiste à considérer que le mouvement de l'objet dans l'image est faible d'une image à l'autre. Ainsi, la pose de l'image courante peut être facilement estimée à partir de la pose de l'image précédent (*eg.* on considère que c'est la même pose), ce qui permet du même coup de prédire la position des primitives 3D du modèle dans l'image (projection à partir de la pose prédite). La mise en correspondance entre primitives peut alors être effectuée par un simple critère de proximité (la primitive 3D est associée à la primitive 2D la plus proche de son projeté dans l'image).

Cette hypothèse est largement utilisée pour le suivi exploitant des primitives de type arêtes 3D tels que Lowe (1987); Harris (1992); Marchand *et al.* (1999); Marchand et Chaumette (2002); Drummond et Cipolla (2002); Comport *et al.* (2003). D'une manière générale, l'approche consiste à extraire les arêtes franches du modèle 3D et à les projeter dans l'image selon la pose prédite. Les arêtes 2D résultantes sont échantillonnées en un ensemble fini de points, et chacun de ces points est associé au point de contour le plus proche situé le long de la droite orthogonale au segment 2D et passant par le-dit point 2D échantillonné. L'ensemble des distances séparant les points échantillonnés au point de contour qui leur est associé représente alors l'erreur de reprojection du modèle dans l'image. Ces solutions se distinguent alors sur la méthode utilisée pour estimer la correction à appliquer à la pose de manière à réduire cette erreur de reprojection.

Notons aussi que l'étape de mise en correspondance des points échantillonnés avec les points de contour est sujette à erreur en raison du problème d'*aperture*. C'est pourquoi ces méthodes sont généralement itératives (le processus de raffinement est relancé à partir de la dernière mise à jour de la pose). La robustesse du processus peut aussi être améliorée en ajoutant, au critère de proximité, des critères de similarité entre l'orientation de l'arête et celle du point de contour associé lors de la mise en correspondance (Bouthemy (1989); Comport *et al.* (2006)), ou en considérant plusieurs hypothèses d'appariement dans le processus d'optimisation de la pose de la caméra Vacchetti *et al.* (2004); Wuest *et al.* (2005); Teuliere *et al.* (2010) (voir la figure 2.5). Par ailleurs, certaines méthodes de suivi d'objet telles que Armstrong et Zisserman (1995); Bleser *et al.* (2005) permettent de limiter les problèmes liés aux données aberrantes, en employant l'algorithme RANSAC proposé par Fischler et Bolles (1981). Dans le même but, Drummond et Cipolla (2002); Vacchetti *et al.* (2004); Comport *et al.* (2006) utilisent un M-estimateur et la méthode des moindres carrés itérativement repondérés (IRLS pour *Iteratively Reweighted Least*

Squares) pour gérer les données aberrantes.

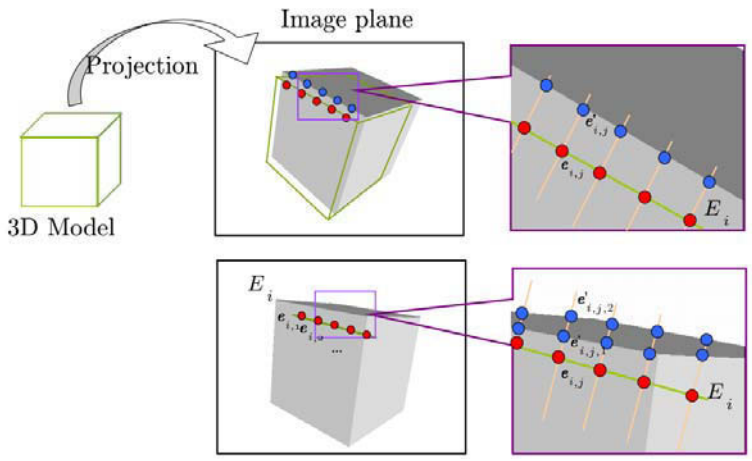


FIGURE 2.5 – Dans les méthodes classiques de suivi utilisant les contours, le modèle est projeté dans le plan de l'image et est échantillonné en plusieurs points. Une recherche est effectuée le long de la normale à l'arête projetée (en haut). Lorsque plusieurs contours sont proches de ces points projetés dans l'image, des ambiguïtés peuvent se produire lorsque la recherche se fait le long de la normale (en bas). Dans ce cas, plusieurs hypothèses peuvent être considérées (extrait de Teuliere *et al.* (2010)).

Ces méthodes présentent donc l'avantage de fournir une estimation précise de la pose de la caméra et d'être robuste aux variations d'aspect de l'objet, que celles-ci soient liées à l'usure de l'objet (*eg.* taches sur l'objet) ou aux variations des conditions d'illumination. Cependant, afin de compenser le faible pouvoir discriminant des primitives utilisées, ces méthodes exploitent généralement l'hypothèse de faible mouvement pour établir les correspondances. Or, la validité de cette hypothèse ne peut généralement pas être garantie dans les applications de RA, ce qui limite donc la robustesse de ces systèmes dans ce contexte.

2.2.2 Méthodes basées sur un modèle 3D photo-géométrique

Les limitations des approches de localisation exploitant un modèle purement géométrique sont donc principalement liées à l'étape de mise en correspondance entre les primitives 3D du modèles avec les primitives 2D de l'image. En effet, si le modèle géométrique permet de prédire la forme qu'auront ces primitives dans l'image (*eg.* des points d'intérêt 2D pour les points 3D, des segments 2D pour des arêtes 3D), cette forme n'est généralement pas suffisamment discriminante pour permettre une association individuelle entre les primitives 2D de l'image et les primitives 3D du modèle. Pour éviter ces problèmes de mise en correspondance de

primitives géométriques, une approche consiste à considérer un modèle 3D de l'objet combinant à la fois sa forme 3D et son apparence, ou modèle photo-géométrique 3D. Ainsi, l'apparence de chaque primitive 3D du modèle peut être comparée à l'apparence des primitives 2D de l'image courante afin de décider de leur mise en correspondance ou non (voir la figure 2.6).

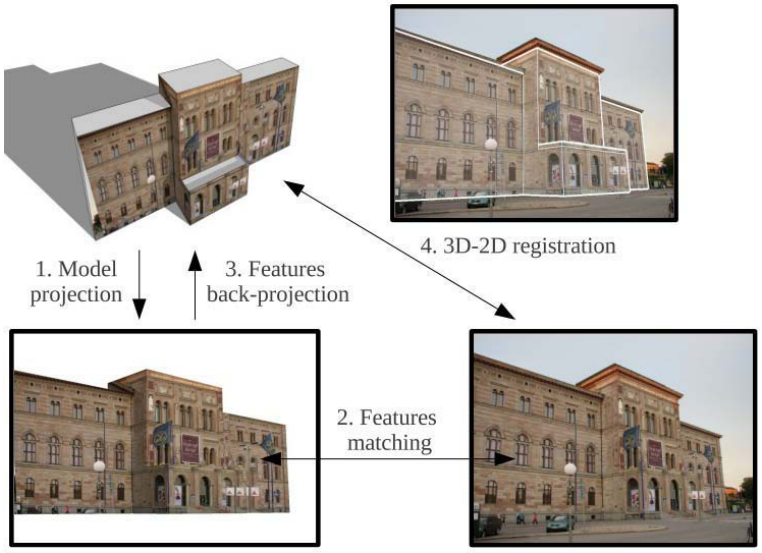


FIGURE 2.6 – Mise en correspondance de l'image courante avec une image de synthèse générée à partir du modèle 3D (description d'une itération de l'algorithme proposé par Simon (2011)).

Ainsi, de nombreux travaux ont proposé de modéliser l'objet d'intérêt à l'aide d'un nuage de points 3D (issu de SfM ou d'un modèle texturé) où l'apparence de chacun de ces points est caractérisée par un ou plusieurs descripteurs locaux (Lowe (2004); Lepetit et Fua (2006); Rothganger *et al.* (2006); Simon (2011)). Ces solutions sont particulièrement robustes puisqu'elles peuvent traiter chaque image indépendamment les unes des autres, évitant ainsi l'hypothèse de faible mouvement par rapport à l'image précédente. Cependant, la précision et la répétabilité de l'extraction de primitives de type point d'intérêt ou région d'intérêt peut alors entrainer des phénomènes de tremblements, ou *jitter*. Pour éviter ce problème, Gordon et Lowe (2006) proposent d'introduire un terme de régularisation venant lisser la trajectoire de la caméra.

Aussi, si la présence d'information d'apparence permet d'accroitre le pouvoir discriminant des primitives à mettre en correspondance, elle permet d'exploiter un nouveau type de primitives. Ainsi, si l'on dispose d'un modèle surfacique 3D texturé, il devient possible de mettre en correspondance les faces de ce modèle avec la surface de l'image leur correspondant. La mise en correspondance est alors exprimée sous la forme d'un problème d'alignement d'images et la pose optimale de

la caméra correspond alors à celle alignant au mieux l'ensemble des images de texture associées aux facettes du modèle 3D de l'objet avec l'image observée par la caméra, ce qui correspond généralement à la pose minimisant la différence de niveaux de gris entre la texture de chaque facette texturée du modèle et la surface de l'image sur laquelle chacune d'entre elles se projette. Cette approche a notamment été utilisée par Benhimane et Malis (2006), ces derniers utilisant l'algorithme ESM (pour *Efficient Second-order Minimization*) pour permettre un traitement en temps réel. Par ailleurs, Caron *et al.* (2012) proposent une solution reposant sur l'information mutuelle qui permet d'accroître la robustesse aux conditions d'illumination au prix d'un temps de traitement plus élevé. Si cette approche présente l'avantage de ne pas nécessiter d'extraction de primitives et donc d'être moins sujette au phénomène de *jitter*, celle-ci nécessite de faibles mouvements entre images consécutives afin de pouvoir prédire une pose initiale pour l'image courante qui soit relativement proche de la solution, et est aussi moins robuste aux occultations partielles de l'objet.

Pour bénéficier des avantages des deux méthodes précédemment citées, Ladikos *et al.* (2007) proposent de combiner les deux approches dans un même processus de localisation. Ainsi, tant que la méthode d'alignement d'images (ESM) fonctionne, celle-ci est utilisée pour obtenir une localisation dépourvue de *jitter*, et quand celle-ci échoue, la méthode de mise en correspondance de points d'intérêt prend le relais pour réinitialiser le processus de suivi par alignement d'images. D'une manière relativement similaire, Wagner *et al.* (2010) proposent d'initialiser/ré-initialiser le suivi à l'aide d'une étape de mise en correspondance de points d'intérêts, puis d'utiliser une technique d'alignement d'images pour suivre les patchs de texture qui entourent chacun de ces points d'intérêt dans les images suivantes. Le suivi étant réalisé sur des régions locales mais sans utiliser un détecteur de points d'intérêt, cette solution permet d'être à la fois moins sensible aux occultations tout en réduisant le phénomène de *jittering* par rapport aux solutions reposant uniquement sur les points d'intérêt. Cette méthode présente aussi l'avantage de réduire fortement les temps de calcul, ce qui est particulièrement intéressant pour l'exploitation de ce type de solution sur des plateformes mobiles de type *smartphone*.

Cependant, notons que l'ensemble des méthodes photo-géométriques précédemment citées reposent exclusivement sur l'apparence encodée dans le modèle de l'objet pour établir les correspondances entre ce dernier et l'image. Ces méthodes sont donc particulièrement sensibles aux variations des conditions d'illumination, ces dernières pouvant faire varier de manière notable l'apparence de l'objet. Pour réduire l'influence de l'illumination, Pressigout et Marchand (2006) proposent de combiner l'approche d'alignement d'images avec une mise en correspondance de primitives purement géométrique, à savoir la mise en correspondance d'arêtes du modèle avec les contours de l'image, mais aussi d'évaluer l'alignement d'images avec des critères moins sensibles aux variations d'illumination (voir la figure 2.7). Si ces approches présentent l'avantage d'être plus robustes aux conditions d'illumination, elles nécessitent cependant de réintroduire l'hypothèse de faible mouvement

pour permettre la mise en correspondance entre arêtes et contours.

FIGURE 2.7 – Méthode de localisation, par rapport à un modèle 3D photo-géométrique, proposée par Pressigout et Marchand (2006). Images pour la première ligne : suivi 3D basé contours, la deuxième ligne : suivi 2D basé sur l'apparence, la troisième ligne : suivi hybride. Seul le suivi hybride permet de suivre correctement l'objet sur toute la séquence malgré les spécularités et les modifications d'aspect de l'objet.

En conclusion, l'introduction de la notion d'apparence dans le modèle 3D de l'objet permet donc de simplifier l'étape de mise en correspondance entre les primitives issues du modèle et celles issues de l'image, mais ceci au prix d'une sensibilité aux conditions d'illumination. De plus, il n'est pas toujours aisé de disposer d'un tel modèle puisque celui-ci doit non seulement représenter la forme de l'objet mais aussi son apparence courante. Ainsi, si un objet a subit des changement d'aspect important en raison de son exploitation (taches, traces d'usure, *etc.*), le modèle doit être mis à jour pour prendre en compte ces changements sous peine d'une localisation moins robuste.

2.3 Localisation par rapport à un modèle 3D avec mise à jour

L'apparence d'un objet pouvant varier dans le temps, ceci en raison de l'usure de celui-ci ou des variations des conditions d'illumination, il est généralement difficilement envisageable de disposer d'un modèle photo-géométrique 3D d'un objet correspondant à l'apparence actuelle de ce-dit objet. Pour répondre à ce problème, une approche consiste à apprendre, ou à mettre à jour, en ligne la modélisation de l'apparence de l'objet, garantissant ainsi l'adéquation du modèle photo-géométrique mis à jour avec l'objet observé.

Ainsi, Bleser *et al.* (2005); Platonov *et al.* (2006); Sourimant *et al.* (2007); Kyrki et Kragic (2011) proposent d'exploiter un modèle 3D purement géométrique et de reconstruire en ligne un nuage de points 3D caractérisés par des descripteurs locaux d'apparence. Ce nuage de points 3D est obtenu à partir des points d'intérêts extraits du flux vidéo en rétro-projetant ces derniers sur le modèle 3D de l'objet. Ainsi, pour chaque point d'intérêt se projetant sur le modèle 3D de l'objet, il est possible d'obtenir sa position 3D ainsi qu'un descripteur local d'apparence. Le nuage de points 3D ainsi obtenu est alors utilisé pour estimer la pose de l'objet dans les images suivantes, poses qui sont alors utilisées pour reconstruire de nouveaux points 3D (voir la figure 2.8).

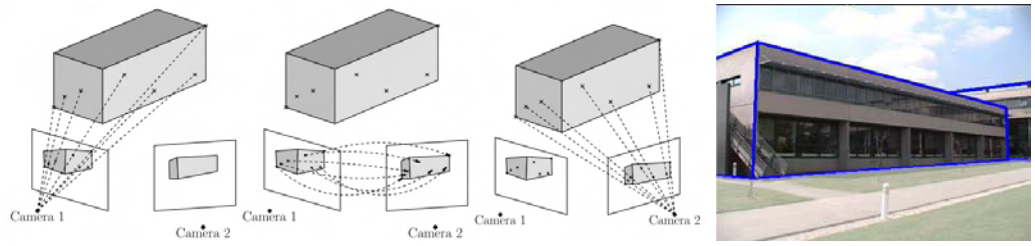


FIGURE 2.8 – Modèle 3D géométrique mis à jour en ligne par une information de texture 3D par la rétroprojection des points 2D texturés sur ce modèle (extrait de Sourimant *et al.* (2007)).

Cette méthode présente un avantage majeur par rapport aux méthodes sans mise à jour. Les descripteurs locaux étant calculés à partir d'images correspondant aux conditions d'éclairage et à l'apparence courante de l'objet, le modèle d'apparence obtenu est ainsi en parfaite adéquation avec la scène observée, évitant ainsi les problèmes de robustesse liées à l'apparence des méthodes sans mise à jour. Néanmoins, le nuage de points 3D étant reconstruit par simple rétro-projection, la précision de celui-ci dépend de la précision du modèle géométrique 3D, de la précision de la pose de la caméra (et en particulier la pose de la première caméra), de la précision de la détection des amers 2D et de l'absence d'occultations. De plus, le processus de localisation et le processus de reconstruction étant invoqués de manière alternée sans jamais remettre en cause les poses des caméras et position des points 3D préalablement estimées, ce type de processus est généralement sujet à une accumulation d'erreurs entraînant une dérive de la localisation, ainsi qu'au phénomène de *jitter*. Notons entre autres, que le processus de localisation doit être initialisé avec une pose extrêmement précise, ce qui peut être une contrainte forte en fonction de l'application considérée.

Pour résoudre ces problèmes, Vacchetti *et al.* proposent tout d'abord de calculer la pose de la caméra non seulement à partir des primitives apprises en ligne, mais aussi à partir de celles issue du modèle initial de l'objet, qu'elles soient géométrique (Vacchetti *et al.* (2004)) ou photo-géométrique (Vacchetti et Lepetit (2004)) (figure

2.9). Ainsi, les primitives 3D issues du modèle initiale permettent d'éviter l'accumulation d'erreurs alors que les primitives apprises en ligne permettent de mieux contraindre l'estimation du mouvement entre deux images. Pour éviter le phénomène de *jittering*, Vacchetti *et al.* (2004) proposent aussi d'appliquer un ajustement de faisceaux sur l'image courante et précédente afin de remettre en cause à la fois la position de la caméra courante et précédente mais aussi la position des points appris en ligne à partir de l'image précédente.

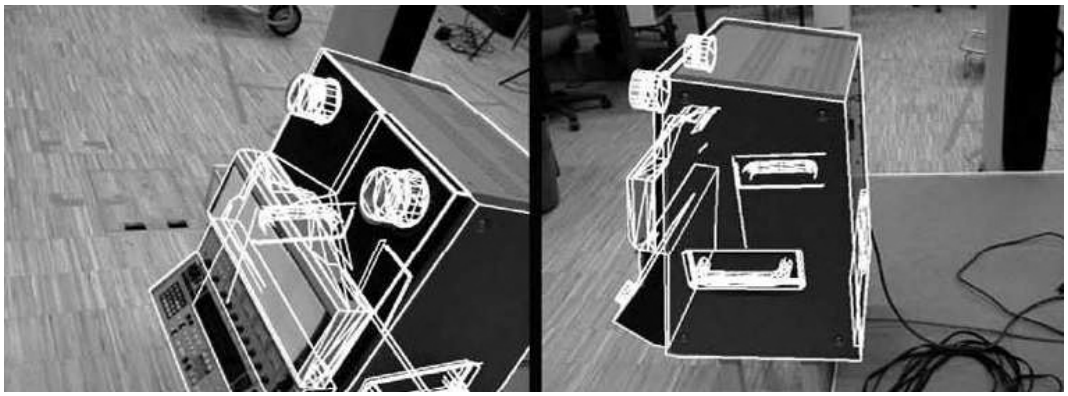


FIGURE 2.9 – Suivi par recalage 3D-2D : travaux réalisés à l'EPFL par Vacchetti et Lepetit (2004). Modèle 3D mis à jour en ligne par une information de texture 3D par la rétroprojection des points 2D texturés sur ce modèle.

En conclusion, les méthodes de localisation par rapport à un modèle photo-géométrique mis à jour permettent donc d'améliorer la robustesse aux conditions d'illumination. Cependant, la pose de la caméra reste estimée uniquement à partir des primitives images correspondant à l'objet, ce qui nécessite que l'objet soit en permanence visible à l'image et couvrant un part importante de celle-ci, et que celui-ci ne soit pas soumis à de trop grandes occultations.

2.4 Localisation en environnement partiellement connu

Les méthodes de localisation par rapport à un modèle 3D, avec ou sans mise à jour, exploitent de l'image uniquement l'information visuelle correspondant à l'objet. Or, si cet objet d'intérêt est statique par rapport au reste de la scène, les éléments inconnus de la scène peuvent apporter des contraintes supplémentaires (contraintes multi-vues à la manière des approches de type SLAM).

Les approches de localisation en environnement partiellement connu consistent donc à exploiter à la fois les contraintes géométriques apportées par le modèle 3D de l'objet d'intérêt et les contraintes de la géométrie multi-vues apportées par l'environnement inconnu pour estimer la pose de la caméra.

Une première solution fut proposée par Bleser *et al.* (2006), celle-ci consistant à initialiser un EKF-SLAM à partir d'une localisation obtenue à partir d'une méthode exploitant un modèle géométrique 3D de l'objet. Plus précisément, Bleser *et al.* (2006) estiment la pose de la caméra par une méthode classique d'alignement des arêtes du modèle avec les contours de l'image. Cette pose est ensuite utilisée pour reconstruire un premier nuage de points 3D en rétro-projetant les points d'intérêt de l'image sur le modèle 3D de l'objet. La première pose étant considérée comme quasi-parfaite, les points 3D de ce nuage initial se voient attribuer une incertitude extrêmement faible. Un processus de EKF-SLAM est alors utilisé pour estimer la pose de la caméra dans les images suivantes. En initiant le processus de EKF-SLAM avec un recalage et une reconstruction exploitant le modèle géométrique de l'objet, Bleser *et al.* (2006) fournissent au processus de EKF-SLAM un repère et une échelle cohérente avec le modèle de l'objet. Cependant, cette solution reste sujette à accumulation d'erreurs dès lors que le point de vue ne permet plus d'observer des points 3D issus de la reconstruction par rétro-projection. De plus, cette reconstruction initiale s'appuie sur une localisation à partir d'une seule vue, ce qui implique potentiellement une erreur non négligeable de la position de la caméra selon l'axe optique. Cette erreur sera alors conservée tout au long du processus puisque la position de la première caméra n'est jamais remise en cause.

Pour éviter ces problèmes, une première solution consiste à alterner le processus de SLAM et une méthode de localisation par rapport à un modèle. Ainsi, Kempter *et al.* (2012) proposent d'utiliser une localisation par rapport à un modèle géométrique 3D lorsque l'objet couvre une grande surface dans l'image, et d'utiliser une méthode de SLAM le reste du temps. Une autre solution (Gay-Bellile *et al.* (2010b,a)) consiste à utiliser un processus de SLAM basé image clé et de supprimer l'accumulation d'erreurs à chaque fois qu'une image clé est mise en correspondance avec des primitives du modèle photo-géométrique 3D. Notons également la méthode de Lothe *et al.* (2010) qui proposent de localiser une caméra mobile dans un environnement urbain en exploitant un modèle géométrique 3D de la ville. Ils utilisent successivement une technique de SLAM puis le modèle pour localiser le véhicule. Afin de réduire la dérive du SLAM, ils proposent de corriger localement la trajectoire de la caméra à chaque intersection. La correction est réalisée par une méthode de type ICP (pour *Iterative Closest Point*) en recalant la reconstruction de points 3D sur le modèle géométrique 3D.

Si ces solutions offrent une meilleure précision de la localisation que les méthodes de type SLAM et une meilleure robustesse par rapport aux méthodes basées modèle (avec ou sans mise à jour), celles-ci restent sous optimales puisqu'elles n'optimisent pas simultanément l'ensemble des éléments de la scène (ie. objet d'intérêt et environnement inconnu) pour l'ensemble des contraintes (contraintes issues du modèle et contraintes multi-vues).

2.5 Discussion et positionnement

Les solutions de localisation en environnement partiellement connu sont celles qui offrent le maximum d'avantages dès lors que l'objet est statique par rapport à la scène. Cependant, les solutions existantes n'exploitent pas simultanément l'ensemble des contraintes disponibles pour garantir une reconstruction de l'environnement inconnu et une localisation de l'objet connu optimales. En effet, les solutions actuelles se limitent à optimiser de manière alternée les contraintes issues du modèle avec celles issues de la géométrie multi-vues Gay-Bellile *et al.* (2010b); Kempter *et al.* (2012), ce qui ne permet pas de garantir que la trajectoire et la reconstruction de la scène estimées soient optimales pour l'ensemble des contraintes.

Dans ce mémoire, nous proposons donc une solution de localisation en environnement partiellement connu reposant sur une algorithme de SLAM intégrant simultanément les contraintes issues d'un modèle 3D de l'objet et les contraintes multi-vues apportées par l'ensemble de la scène. Ainsi, lorsque l'objet est bien visible dans l'image, les contraintes issues du modèle garantissent une localisation précise et donc une reconstruction précise de l'environnement. Cet environnement étant cartographié avec précision dans le même repère et à la même échelle que l'objet d'intérêt, ce dernier peut alors être localisé à partir de ce seul environnement, c'est à dire même si l'objet est petit, occulté ou en dehors du champ de vision.

CHAPITRE

3

Ajustement de faisceaux contraint : un cadre d'unification des méthodes de localisation

Le but de la thèse est de proposer une solution unique pour la localisation permettant de gérer différents types d'objets et d'environnements. L'approche repose sur une méthode de type SLAM que l'on a modifiée pour y intégrer des contraintes liées à la connaissance a priori du modèle (géométrique ou photo-géométrique) d'un objet d'intérêt de la scène observée. On parle dans ce cas de localisation d'une caméra en environnement partiellement connu. Ce chapitre discute des méthodes de localisation existantes et motive l'approche proposée qui repose sur un ajustement de faisceaux contraint intégré dans une méthode de type Keyframe-based SLAM.

3.1 Présentation de la solution proposée

3.1.1 Principe général

Nous avons vu dans le chapitre précédent que l'approche de localisation en environnement partiellement connu est celle qui permet d'exploiter l'ensemble des informations disponibles (les contraintes de l'environnement et celles liées au modèle). Cependant les solutions proposées jusqu'alors comme Lothe *et al.* (2010) ou Kempter *et al.* (2012) n'offrent pas la robustesse et la précision attendue du fait qu'elles se limitent à alterner des méthodes de suivi basé modèle et des méthodes de type SLAM. Pour remédier à ce problème, nous proposons dans cette thèse une unification des méthodes de localisation basées modèle avec un SLAM.

De notre point de vue, les méthodes de localisation basées modèle comme celles de type SLAM peuvent être décomposées en deux processus : un processus de cartographie et un processus de localisation. Ces méthodes se distinguent néanmoins par :

- ▷ le fait que le processus de cartographie est réalisé soit en ligne (SLAM), soit hors ligne (méthodes basées modèle), soit en ligne et hors ligne (méthodes basées modèle avec mise à jour) ;
- ▷ la nature des primitives traitées par ces deux processus : primitives issues de l'objet d'intérêt (méthodes basées modèle avec et sans mise à jour) ou primitives issues de l'ensemble de l'environnement (SLAM).

Pour unifier les deux approches le localisation que sont les méthodes basées modèle et les méthodes de type SLAM l'objectif est de proposer :

- ▷ un processus de modélisation permettant d'intégrer à la fois des données acquises hors ligne (modèle 3D de l'objet) comme en ligne (objet et environnement) ;
- ▷ un processus de localisation exploitant à la fois les primitives cartographiées de l'objet comme de l'environnement.

Cette nouvelle méthode de localisation correspond à un **SLAM contraint** par l'information absolue issue d'un modèle 3D d'un objet d'intérêt de la scène. En effet, sous l'hypothèse que l'objet est statique dans la scène, les primitives reconstruites par le SLAM et celles extraites du modèle permettent d'estimer les poses de la caméra. Comme le résume le tableau 3.1, cette solution présente l'intérêt de combiner les avantages des méthodes SLAM et basées modèle tout en levant leurs limites. En effet, la reconstruction en ligne des points de l'environnement permet d'améliorer la robustesse des méthodes basées modèle. Inversement, inclure une information absolue d'un objet statique dans le SLAM permet de réduire la dérive et de fixer le facteur d'échelle et le repère initial. Mais pour cela, il faut que les processus de reconstruction et de localisation soient optimaux vis-à-vis des différents types de données (objet/environnement et hors ligne/en ligne). Dans les paragraphes suivants, nous expliquerons pourquoi l'utilisation d'un ajustement de faisceaux contraint permet d'obtenir une solution optimale puis nous présenterons différentes manières de formuler cet ajustement de faisceau contraint.

3.1.2 Motivations pour l'ajustement de faisceaux contraint

La principale difficulté est d'intégrer l'information du modèle de **manière optimale** dans le processus SLAM (voir la figure 1.9). Aussi, s'il est possible d'introduire les contraintes liées au modèle de l'objet lors de chacune des étapes du SLAM, toutes ces solutions ne sont pas équivalentes.

Ainsi, intégrer le modèle dans l'initialisation du SLAM (Bleser *et al.* (2006)) permet d'obtenir une reconstruction à l'échelle et par rapport au repère absolu fixé

	Environnement sans modèle	Modèle sans mise à jour	Modèle avec mise à jour	Environnement avec modèle	Notre méthode
dérive	- -	++	++	-	+ +
stabilité	++	- -	-	+ +	+ +
précision	- -	++	++	-	+ +
éclairage [a]	++	- - / ++	+	+ +	+ +
occultation	++	- -	- -	++	+ +
géoréférencé	- -	+ +	+ +	+ +	+ +
objet dynamique [b]		+ +	+ +	- -	- -

a. Pour la méthode de suivi basé modèle sans mise à jour, la notation dépend dans ce cas de la nature du modèle. Deux plus sont attribués à la méthode si le modèle est géométrique et deux moins sont attribués si le modèle est photo-géométrique.

b. Pour l'aspect objet dynamique le SLAM n'est pas noté car il n'y a pas de notion d'objet pour ce type de méthodes. Par ailleurs il est à noter que pour la méthode proposée des solutions pour ce problème peuvent être envisagées et elles seront mentionnées en détails dans les perspectives de ce mémoire.

TABLE 3.1 – Comparaison des méthodes de l'état de l'art et positionnement de nos travaux. Les méthodes de localisation sont comparées entre elles en termes de dérive, de stabilité, de précision, de robustesse à des changements d'éclairage, de robustesse aux occultations. De plus elles sont évaluées sur le géoréférencement de la localisation et sur leur capacité à traiter le problème de localisation par rapport à un objet dynamique.

sur le modèle. Cependant, ceci n'est pas optimal car l'information du modèle n'est plus utilisée par la suite, donc une erreur au départ est conservée voire amplifiée dans la suite du processus de localisation. De même, intégrer les contraintes du modèle lors de la localisation ne garantit pas une reconstruction optimale, et inversement contraindre la localisation avec le modèle de l'objet ne garantit pas une localisation optimale. De plus, dans chacun de ces deux cas, si les contraintes ne sont pas maintenues lors de l'étapes d'ajustement de faisceaux, rien n'assure qu'elles soient encore vérifiées à son issue.

Par contre, l'introduction des contraintes du modèle dans l'ajustement de faisceaux permet une estimation optimale puisque ce dernier minimise **simultanément** dans un critère unique les contraintes géométriques fournies par le modèle et les contraintes multi-vues. Ainsi, il assure à la fois une reconstruction et une localisation optimale vis à vis des contraintes géométriques issues du modèle et de la géométrie multi-vues.

Par ailleurs, il est possible de combiner plusieurs types de contraintes et de primitives, dans l'ajustement de faisceaux dans le mesure où les critères sont exprimés dans la même unité. Le SLAM contraint est alors suffisamment flexible pour per-

mettre la combinaison de différentes contraintes et de primitives, à travers un ajustement de faisceaux avec contraintes multiples. L'intérêt de combiner différentes contraintes, permet de gérer le suivi d'un plus large panel d'objets (texturés, peu texturés, cylindrique, *etc.*).

C'est pourquoi nous avons choisi d'intégrer les contraintes liées au modèle de l'objet au niveau de l'ajustement de faisceaux. La figure 3.1 présente la structure globale du SLAM contraint proposé.

A noter que l'ajout des contraintes n'est pas réalisé à chaque image, celles-ci sont intégrées dans l'ajustement de faisceaux réalisé uniquement aux images clés. En effet, si introduire le modèle dans l'ensemble des étapes du SLAM pourrait sembler idéal, nous montrerons dans les chapitres suivants que limiter les contraintes à l'ajustement de faisceaux est suffisant et permet de garantir de meilleures performances. De plus, le formalisme que nous proposons pour l'ajout des contraintes permet de conserver la structure creuse de l'ajustement de faisceaux classique (voir les sections 4.4 et 5.6). Ceci permet d'obtenir des performances temps réel vidéo. Rappelons que l'ajustement de faisceaux peut être réalisé de manière locale comme globale. Il est également possible d'appliquer aisément les contraintes décrites ici, à l'ensemble de la reconstruction 3D de la séquence.

3.1.3 Différentes façons d'introduire les contraintes dans l'ajustement de faisceaux

La stratégie utilisée pour introduire les contraintes dans l'ajustement de faisceaux ne peut être la même quelque soit la nature de l'objet et de son modèle. C'est pourquoi, afin de gérer différents types de modèles et d'objets, nous proposons dans le cadre de cette thèse deux solutions :

- ▷ Unification SLAM et localisation par rapport à un modèle 3D connu.
- ▷ Unification SLAM et localisation par rapport à un modèle 3D avec mise à jour.

Unification SLAM et localisation par rapport à un modèle 3D connu. Nous préconisons cette approche lorsque la précision du modèle 3D permet d'associer des primitives 3D (points, segments, *etc.*) extraites du modèle avec des primitives 2D (points 2D, segments 2D, contours 2D) détectées dans les images. Dans ce cas nous proposons de reconstruire avec le SLAM des primitives qui ne sont pas présentes dans le modèle 3D. Cette méthode estime alors les poses de la caméra conjointement avec les primitives reconstruites en ligne par le SLAM et celles du modèle. Quand l'objet n'est plus visible, les poses sont estimées uniquement par les primitives reconstruites par le SLAM. La figure 3.2 décrit un exemple de scène 3D associée à cette méthode : la scène est composée d'un ensemble de primitives 3D reconstruites par le SLAM et un ensemble de primitives 3D fournies par le modèle.

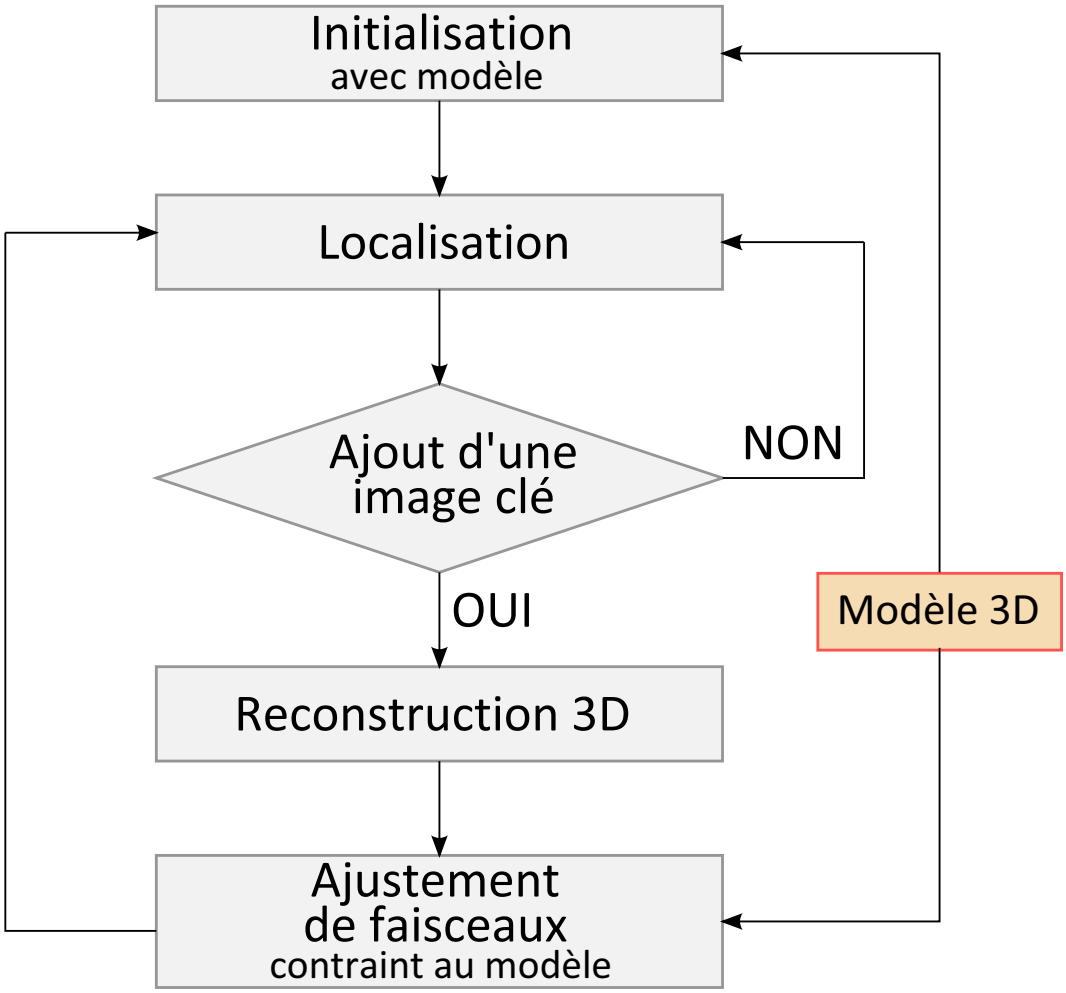


FIGURE 3.1 – Processus SLAM contraint par un modèle 3D. Dans l'approche proposée, l'information liée au modèle est directement intégrée dans l'ajustement de faisceaux ainsi que dans la phase d'initialisation.

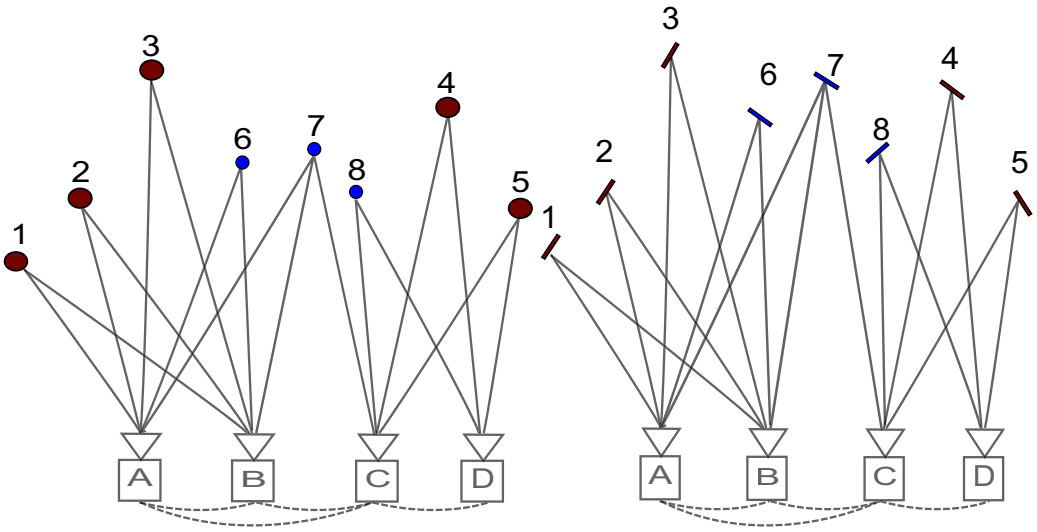


FIGURE 3.2 – Schéma décrivant l'unification SLAM et localisation par rapport à un modèle 3D connu : (a) pour un nuage de points 3D et (b) pour un ensemble de segments 3D issus d'un SLAM. En rouge, les primitives 3D de la partie inconnue de l'environnement ; en bleu, les primitives 3D associées à la partie connue de l'environnement. Les lignes noires indiquent l'observabilité d'une primitive 3D pour chacune des poses de la caméra en mouvement (de A à D).

Unification SLAM et localisation par rapport à un modèle 3D avec mise à jour. Dans certains cas il n'est pas possible d'associer des primitives 3D du modèle avec des primitives 2D détectées dans les images (modèle surfacique grossier sous forme de maillage). Les primitives 3D reconstruites (points 3D, segments 3D, *etc.*) peuvent alors être associées aux plans du modèle, et être contraintes à appartenir à ce dernier. La scène est alors composée uniquement d'un modèle surfacique de l'objet et de primitives 3D reconstruites par le SLAM. Parmi celles-ci, certaines sont considérées comme appartenant à l'environnement et les autres au modèle. Ces dernières venant compléter le modèle surfacique, cela peut être vu comme une mise à jour du modèle. La figure 3.3 décrit un exemple de scène 3D associée à cette méthode. De plus cette figure illustre le fait que la géométrie du modèle contraint non seulement la trajectoire de la caméra mais aussi la reconstruction des primitives. A noter que contrairement à la méthode d'unification précédente, les primitives 3D reconstruites et associées au modèle sont contraintes explicitement par le modèle.

Introduction au pSLAM et sSLAM. Les primitives utilisées pour chacune des étapes principales du SLAM (initialisation, reconstruction, localisation, optimisation), peuvent être des points d'intérêt (Mouragnon *et al.* (2006)), des segments de droites (Smith *et al.* (2006)), des courbes (Rao (2012)), des plans (Servant

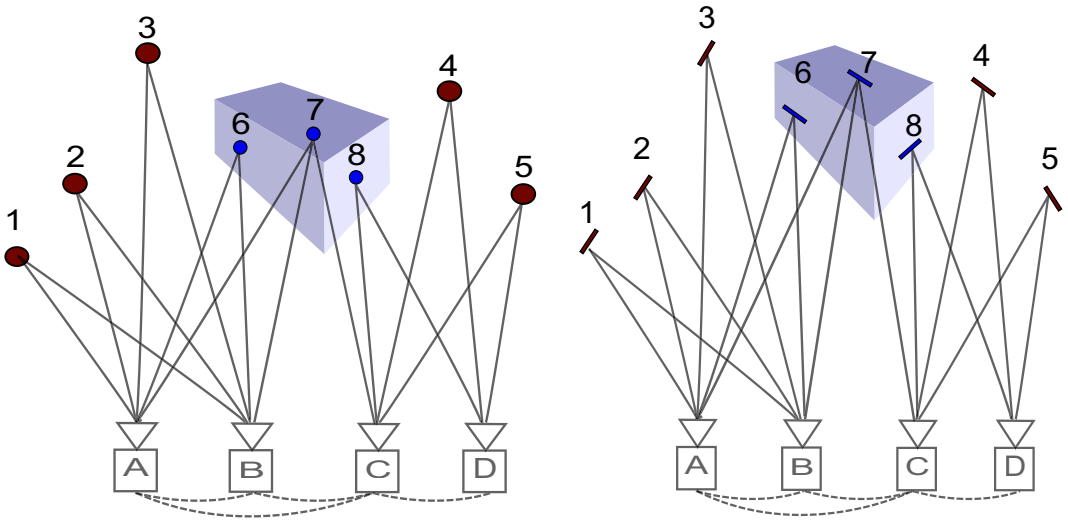


FIGURE 3.3 – Schéma décrivant l'unification SLAM et localisation avec mise à jour du modèle pour des primitives de type points ou segments. En rouge, les primitives 3D de la partie inconnue de l'environnement ; en bleu, les primitives 3D associées à la partie connue de l'environnement. Les lignes noires indiquent l'observabilité d'une primitive 3D à partir de chacune des poses de la caméra en mouvement (de A à D).

et al. (2008)), ou la combinaison de plusieurs types de primitives (Klein et Murray (2008)). Nous proposons par la suite un formalisme de SLAM contraint pour les primitives de type points ou de type segments. En effet, pour pouvoir gérer un large panel d'environnements, il est important de proposer un même cadre de résolution pour ces deux types de primitives. Dans la suite nous utiliserons les appellations suivantes :

- ▷ pSLAM, le SLAM utilisant comme primitives des points d'intérêt,
- ▷ sSLAM, le SLAM utilisant comme primitives des segments.

3.2 Formalisme de l'ajustement de faisceaux contraint

3.2.1 Une fonction de coût multi-termes

L'optimisation avec un ajustement de faisceaux classique consiste à minimiser la somme des erreurs de reprojection, à savoir la somme des distances entre des primitives 3D projetées dans l'image et des primitives 2D détectées et associées (voir la section 1.3.5). Dans l'ajustement de faisceaux contraint proposé, la fonction de coût à minimiser est toujours la somme de ces erreurs de reprojection, cependant elle est composée de deux types de termes :

- ▷ un terme lié aux résidus des primitives de l'environnement ;
- ▷ un terme lié aux résidus des primitives, extraites du modèle ou reconstruites et contraintes au modèle.

Il en résulte une fonction de coût composée de la partie inconnue (c'est-à-dire sans contrainte au modèle) et de la partie connue (c'est-à-dire avec contraintes au modèle) de l'environnement :

$$\mathcal{E} = \mathcal{E}_E + \lambda \mathcal{E}_M, \tag{3.1}$$

où $\mathcal{E}_E$, $\mathcal{E}_M$ sont respectivement les termes associés à la partie inconnue et à la partie connue de l'environnement, et λ est le poids qui contrôle l'influence de chaque terme.

L'ajustement de faisceaux contraint ainsi proposé requiert des étapes supplémentaires comparé à l'ajustement de faisceaux « classique ». En effet, pour introduire la contrainte basée modèle, les primitives doivent être associées au modèle avec une gestion des mauvaises associations. De plus, comme l'illustre l'équation (3.1) nous sommes dans le cas d'une résolution d'un problème bi-objectif. Il requiert la gestion de l'influence de chacun des deux types de terme $\mathcal{E}_E$ et $\mathcal{E}_M$ durant sa minimisation. Nous décrivons dans les sections 3.2.3, 3.2.4 et 3.2.5 les étapes additionnelles nécessaires pour le processus d'ajustement de faisceaux contraint.

Nous rappelons que les contraintes au modèle sont formulées différemment en fonction de la nature des primitives prises en compte : soit nous contraignons les primitives reconstruites à appartenir au modèle, soit nous contraignons les primitives existantes reprojetées à s'aligner avec celles extraites des images. Nous détaillons dans la section 3.2.2 leurs formulations pour les primitives suivantes : points et segments 3D.

3.2.2 Deux types de contraintes

3.2.2.1 Ajustement de faisceaux contraint par projection d'un modèle 3D

Il est possible de contraindre l'ajustement de faisceaux du SLAM par les contraintes fournies par la carte 3D reconstruite préalablement (modèle 3D). Dans le cas de l'unification SLAM et localisation par rapport à un modèle 3D connu, ces primitives extraites du modèle sont considérées sans erreur, leurs positions ne sont pas optimisées dans le SLAM. La contrainte se traduit alors par le fait qu'un point ou un segment 3D extrait du modèle possède zéro degré de liberté. Ainsi le modèle est uniquement utilisé pour contraindre la trajectoire de la caméra. Dans l'équation (3.1) le terme $\mathcal{E}_M$ correspond dans ce cas aux primitives extraites du modèle. Ces primitives contraignent directement la trajectoire de la caméra et par conséquent, elles contraignent indirectement les primitives reconstruites en ligne par le SLAM. Le tableau 3.2 résume les degrés de liberté dans l'ajustement de faisceaux des différentes primitives issues de l'environnement ou du modèle.

Primitives reconstruites pSLAM ($\mathcal{E}_E$)	Primitives pré-reconstruites Modèle 3D ($\mathcal{E}_M$)	
Points 3D	Points 3D	Segments 3D
3-DoF	0-DoF	0-DoF

TABLE 3.2 – **Unification SLAM et localisation basé modèle.** Ajustement de faisceaux contraint par un modèle 3D connu dans le cas du pSLAM. Les contraintes sont exprimées en fonction du nombre de degrés de liberté (DoF pour *Degree of Freedom*) attribué aux différentes primitives.

3.2.2.2 Ajustement de faisceaux contraint aux plans d'un modèle 3D géométrique

Dans le cas de l'unification entre le SLAM et la localisation avec mise à jour du modèle, la contrainte se traduit par le nombre de degrés de liberté accordé à la primitive 3D reconstruite. Par exemple, un point 3D reconstruit qui n'est pas associé au modèle possède 3 degrés de liberté. Un point 3D reconstruit et associé à un plan du modèle ne possède que 2 degrés de liberté. Un segment 3D reconstruit non associé au modèle possède 4 degrés de liberté [1], alors qu'un segment 3D reconstruit, associé à un plan du modèle, possède 2 degrés de liberté. Dans l'équation (3.1) le terme $\mathcal{E}_M$ correspond dans ce cas aux primitives reconstruites par le SLAM et associées au modèle. Le tableau 3.3 résume les degrés de liberté dans l'ajustement de faisceaux des différentes primitives reconstruites par le SLAM, en fonction de leur association à un plan du modèle ou à l'environnement. Notons qu'un modèle 3D géométrique est nécessaire mais cela ne limite pas cette approche à des objets polyédriques car une surface gauche peut aussi être représentée par un maillage (c'est à dire un ensemble de triangles).

	Primitives reconstruites	
	Associées à l'environnement ($\mathcal{E}_E$)	Associées aux plans du modèle ($\mathcal{E}_M$)
pSLAM	3-DoF	2-DoF
sSLAM	4-DoF	2-DoF

TABLE 3.3 – **Unification SLAM et localisation avec mise à jour du modèle.** Ajustement de faisceaux contraint aux plans d'un modèle 3D géométrique, dans les cas pSLAM et sSLAM. Les contraintes sont exprimées en fonction du nombre de degrés de liberté (DoF pour *Degree of Freedom*) attribué aux différentes primitives.

1. Nous décrivons en chapitre 5 la représentation choisie pour les segments 3D. Pour des raisons de stabilité de l'optimisation nous avons choisi une représentation minimale de 4 paramètres.

3.2.2.3 Ajustement de faisceaux avec contraintes multiples

Les deux cas d'unification distincts décrits précédemment et illustrés par les figures 3.2 et 3.3 peuvent être réalisés simultanément. La combinaison de différentes contraintes ainsi que la reconstruction de primitives de différentes natures peuvent être envisageables. Le tableau 3.4 résume l'ensemble des différentes contraintes au modèle envisagées dans cette thèse pour le pSLAM ou le sSLAM.

Primitives reconstruites			Primitives pré-reconstruites Modèle 3D	
	Associées à l'environnement	Associées aux plans	Points 3D	Segments 3D
pSLAM	3-DoF	2-DoF	0-DoF	0-DoF
sSLAM	4-DoF	2-DoF	0-DoF	0-DoF
terme associé [a]	$\mathcal{E}_E$	$\mathcal{E}_M$		

a. Il s'agit du terme associé dans la fonction de coût décrite par l'équation (3.1).

TABLE 3.4 – **Unification des différentes méthodes de localisation par contraintes multiples.** Tableau récapitulatif des différents types de contraintes proposées pour le pSLAM et le sSLAM. Les contraintes sont exprimées en fonction du nombre de degrés de liberté (DoF pour *Degree of Freedom*) attribué aux différentes primitives.

3.2.3 Phase d'association des données au modèle

Les deux approches d'unifications présentées précédemment se distinguent principalement par la façon dont les primitives sont mises en correspondance avec le modèle. Les contraintes sont des contraintes géométriques qui s'appliquent différemment en fonction de la nature de l'association réalisée. Dans un cas, les contraintes s'appliquent implicitement à la trajectoire de la caméra et aux primitives de l'environnement par mises en correspondance 3D-2D (voir les sections 4.2.3 et 4.3.3) entre les primitives du modèle et les primitives images. Dans l'autre cas, les contraintes s'appliquent explicitement aux primitives 3D (points 3D ou segments 3D) associées au modèle et implicitement aux primitives de l'environnement ainsi qu'à la trajectoire de la caméra. Les primitives 3D reconstruites sont pour cela associées aux plans du modèle 3D, nous parlerons alors de mises en correspondances 3D-3D (voir la section 5.4).

3.2.4 Estimation robuste

Un mauvais recalage initial (entre le repère monde et le repère de l'objet) et la dérive inhérente aux algorithmes de type SLAM peuvent introduire de mauvaises associations des primitives au modèle, perturbant la convergence du processus d'optimisation. Pour gérer ces associations aberrantes une estimation robuste est effectuée avec le M-estimateur de Geman-McClure $\rho(r, c) : \mathbb{R} \rightarrow [0 \cdot \cdot 1]$

$$\rho(r, c) = \frac{r^2}{r^2 + c^2}, \tag{3.2}$$

où r est l'erreur résiduelle d'une des fonctions de coût décrites dans les chapitres suivants et qui s'apparente à l'erreur de reprojection décrite par l'équation (1.51), et c le seuil de rejet. Ce dernier est estimé automatiquement en utilisant l'écart absolu à la médiane des erreurs (MAD pour *Median Absolute Deviation*) tel que $c = médiane(\mathbf{r}) + 1.4826\text{MAD}(\mathbf{r})$ où $\mathbf{r}$ est un vecteur concaténant les résidus d'une des fonctions de coût. Notons que le MAD suppose une distribution normale des résidus. Il est défini par l'équation suivante :

$$\text{MAD} = médiane_i(|\mathbf{r}_i - médiane_j(\mathbf{r}_j)|). \tag{3.3}$$

L'équation (3.1) se réécrit alors de la manière suivante avec $\lambda = 1$:

$$\mathcal{E} = \rho(\mathcal{E}_E, c_1) + \rho(\mathcal{E}_M, c_2). \tag{3.4}$$

Dans tout le reste du document nous considérerons toujours $\lambda = 1$ sans traiter le problème du choix de cette valeur. Nous préciserons en section 3.2.5 comment sont gérés les deux termes c_1 et c_2. Par ailleurs, notons dans l'équation (3.4) que le M-estimateur est appliqué aux deux termes de la fonction car les résidus sont normalisés.

3.2.5 Pondération de l'ajustement de faisceaux contraint

Approches existantes pour la pondération de deux fonctions de coût

L'une des grandes difficultés dans la minimisation d'un problème bi-objectif est de contrôler l'influence de chaque terme qui parfois n'ont pas la même unité. Dans cette thèse nous avons utilisé uniquement des contraintes exprimées en pixels pour faciliter la pondération mais néanmoins, en pratique, les deux fonctions n'ont pas nécessairement le même ordre de grandeur. Dans notre cas nous avons constaté expérimentalement qu'elles n'ont pas forcement le même ordre de grandeur : les erreurs résiduelles associées à la partie connue de l'environnement sont la plupart du temps plus élevées (voir la section 4.5.1.1).

Généralement, un facteur λ est introduit de façon à pondérer ces deux fonctions (Horn et Schunck (1981); Chui et Rangarajan (2003); Modersitzki (2004);

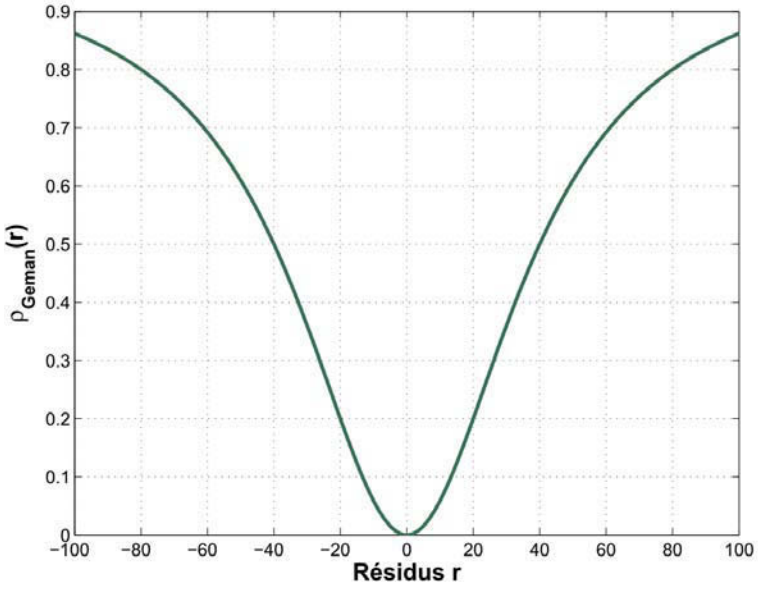


FIGURE 3.4 – Représente ce que devient la contribution quadratique des erreurs en utilisant le M-estimateur de Geman-McClure.

Pilet *et al.* (2005); Michot *et al.* (2010)). La principale difficulté lorsqu'on utilise ce type d'approches est de déterminer le facteur λ qui fournit la solution optimale. Déterminer λ est d'autant plus problématique que la meilleure valeur de ce facteur est généralement fortement dépendante de la séquence traitée (Bartoli *et al.* (2008)). En pratique, ce paramètre est généralement fixé manuellement. Notons cependant que certaines méthodes tendent à l'estimer automatiquement. On peut citer notamment la validation croisée (Farenzena *et al.* (2008)) et plus récemment la méthode basée sur les L-Curve (Michot *et al.* (2010)). Néanmoins, les temps de calcul nécessaires à l'estimation automatique de λ deviennent rapidement extrêmement importants lorsque le nombre de paramètres à optimiser est élevé.

Afin de pondérer les deux termes, nous proposons dans cette thèse, une solution originale basée sur un choix particulier du seuil de rejet de l'estimateur robuste. Il s'agit d'un seuil adaptatif qui est réestimé à chaque ajustement de faisceaux.

Pondération avec l'estimateur robuste

Nous proposons ici une alternative plus simple : l'influence de chaque terme est directement contrôlée en utilisant le seuil de rejet de l'estimateur robuste. Nous avons vu précédemment qu'un estimateur robuste est utilisé pour gérer les mauvaises associations primitive-modèle, pour les fonctions de coût de la partie connue de l'environnement. Ainsi en appliquant également l'estimateur robuste de Geman-McClure aux fonctions de coût de la partie inconnue de l'environnement, tous les résidus sont normalisés. Nous proposons alors différentes fonctions de coût de la

forme de l'équation (3.4), qui seront décrites dans les chapitres suivants.

Il existe alors plusieurs possibilités pour contrôler l'influence de chaque terme avec le seuil de rejet. Nous en avons explorées trois :

- ▷ combinaison 1 : $c_1 = c_{Env}$ et $c_2 = c_{Modele}$
- ▷ combinaison 2 : $c_1 = c_2 = c_{All}$
- ▷ combinaison 3 : $c_1 = c_2 = c_{Max} = \max(c_{Env}, c_{Modele})$

où c_{Modele} est le seuil de rejet estimé sur les résidus du modèle, c'est-à-dire le terme $\mathcal{E}_M$ dans l'équation (3.4). c_{Env} est le seuil estimé sur les résidus de la partie inconnue de l'environnement, c'est-à-dire le terme $\mathcal{E}_E$ dans l'équation (3.4). c_{All} est estimé sur l'ensemble des résidus. Pour la combinaison 1, il y a un seuil de rejet associé à chaque terme alors que pour les combinaisons 2 et 3 un seul seuil est déterminé. La différence est que pour la combinaison 2, il est évalué sur l'ensemble des résidus alors que pour la combinaison 3, seuls les résidus associés à la partie dont les résidus sont les plus élevés ont contribué à son estimation. La combinaison 2 considère que ces deux types de résidus ont le même ordre de grandeur. Les combinaisons 1 et 3 font au contraire l'hypothèse que ces deux types de résidus ont des ordres de grandeur différents. La combinaison 1 traite les deux parties de l'environnement de manière identique alors que la combinaison 3 favorise la partie dont les résidus sont les plus élevés, au cours du processus d'optimisation. Dans ce dernier cas la plupart des points de l'autre partie sont alors conservés du fait de leur valeur plus faible ce qui garantie que les contraintes impliquées par un grand nombre des primitives restent vérifiées. Ces trois combinaisons sont évaluées sur des données de synthèse dans les chapitres 4 et 5.

3.3 Initialisation du SLAM avec un modèle 3D

Le SLAM contraint étant une solution de suivi 3D, celui-ci nécessite une étape d'initialisation fournissant la pose de la caméra pour la première image. Par ailleurs lorsque les primitives du modèle ne sont pas de la même nature que celles reconstruites par le SLAM, une première reconstruction 3D de primitives compatible avec le SLAM doit être obtenue.

A la différence de la solution de Bleser *et al.* (2006), notre solution de SLAM contraint ne nécessite pas que cette estimation initiale de la pose de la première caméra et de la cartographie soit extrêmement précise puisque l'ajustement de faisceaux contraint permet de corriger d'éventuelles erreurs d'initialisation. Ceci permet d'envisager un plus large panel de méthodes d'initialisation et donc de choisir la plus adaptée en fonction du contexte applicatif.

Dans la section 3.3.1, nous présentons une liste non-exhaustive de solution pour l'estimation de pose initiale compatible avec notre approche. Puis, dans la section 3.3.2, nous présentons la solution que nous utilisons pour établir la carte 3D initiale du SLAM.

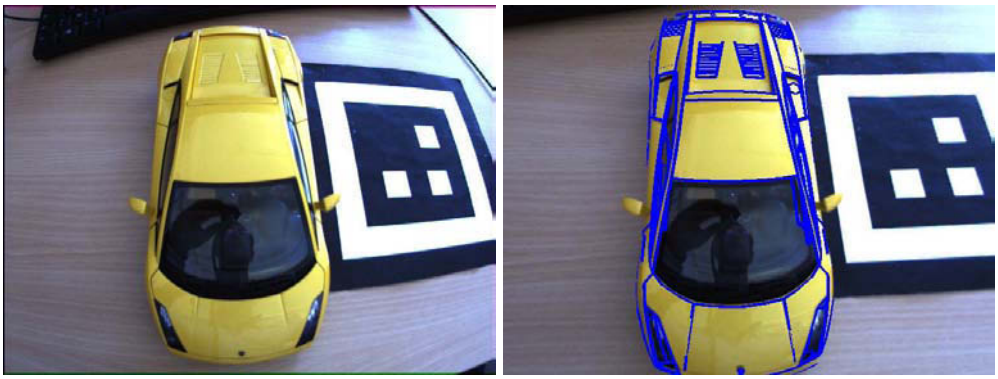


FIGURE 3.5 – Initialisation du SLAM par cible codé : calcul de la première pose de la caméra permettant de recaler dans la première image le modèle 3D sur l'objet d'intérêt.

3.3.1 Localisation initiale contrainte au modèle

Cette étape concerne le calcule de la pose initiale (première pose) c'est-à-dire la position et l'orientation de la caméra par rapport à un repère fixé sur l'objet. Plusieurs méthodes peuvent être utilisées pour cette étape. Nous allons par la suite en présenter certaines plus ou moins automatiques.

Utilisation d'une cible codée localisée par rapport à l'objet Il est possible d'effectuer le recalage initial en positionnant une cible codée géolocalisée par rapport au modèle de l'objet d'intérêt. L'intérêt de la cible codée est qu'elle est facilement détectable dans les images. Cette cible comprend des points d'intérêt, dont les coordonnées 3D sont connues. Un calcule de pose (par mise en correspondance de points 3D-2D) avec l'algorithme de Dementhon et Davis (1995) est réalisé pour estimer la première pose de la caméra (voir la figure 3.5). Une méthode similaire peut être réalisée avec l'algorithme proposé par Ababsa et Mallem (2006) qui utilise un calcul de pose par mise en correspondance 3D-2D de segments.

Semi automatique Bleser *et al.* (2006) ont proposé la méthode semi-automatique qui suit : à partir d'une pose approximative, il applique une méthode de recalage basé sur les contours du modèle pour raffiner celle-ci sur l'image courante. Cette méthode nécessite que l'utilisateur déplace la caméra pour recaler, dans l'image, l'objet avec le modèle qui est reprojeté à partir d'une certaine pose prédéfinie. En pratique cette méthode est difficile à mettre en œuvre car l'utilisateur doit recaler dans l'image le modèle sur l'objet en bougeant, à la main, la caméra (6 degrés de liberté) de manière à atteindre la pose prédéfinie. A noter qu'une alternative peut consister à poser une cible codée sur le sol pour fixer 3 degrés de liberté. L'utilisateur peut ainsi plus facilement recaler (les 3 degrés de liberté restants) le modèle sur

l'objet dans l'image, à condition que la cible soit visible. Une fois que le modèle est approximativement recalé sur l'objet dans l'image, un algorithme de raffinement basé sur les contours du modèle est réalisé, comme dans Bleser *et al.* (2006). A noter que le principal défaut de ces deux méthodes et qu'elles ne permettent pas une localisation initiale à partir d'une pose arbitraire. Cette initialisation n'est possible que pour la pose prédéfinie.

Automatique Une initialisation automatique peut également être envisagée sous certaines conditions. En effet, si l'objet est suffisamment texturé et qu'un modèle de texture de l'objet est disponible, il est possible d'utiliser la relocalisation après avoir appris différents points de vue. Par exemple Platonov *et al.* (2006) utilisent pour l'initialisation, un ensemble d'images référencées hors ligne. Dans leur méthode, l'image de référence qui ressemble le plus à l'image courante est choisie. Les points de l'image courante qui correspondent aux points connus de l'image de référence ainsi sélectionnée sont suivis par la méthode KLT (Tomasi et Kanade (1991)). La pose de la camera est ainsi estimée en se basant sur la position de ces points 3D de référence ainsi que leurs positions 2D de l'image courante. Noter que la recherche peut être réalisée de manière efficace en utilisant un arbre de vocabulaire (Nistér et Stewenius (2006)).

La méthode précédente basée sur la mise en correspondance 3D-2D de points d'intérêt, considère qu'un nombre suffisant de points a été détecté et identifié dans l'image. Par conséquent, ces méthodes ne peuvent être appliquées aux scènes, où les lignes sont nombreuses, mais où l'information de texture est suffisamment riche. Si l'objet n'est pas texturé et que l'on ne possède pas de modèle de texture associé à celui-ci, la méthode de recherche exhaustive, proposée par Prisacariu et Reid (2011) peut être envisagée. La première pose de la caméra est calculée en recherchant la pose assurant la meilleure adéquation entre les contours projetés du modèle 3D et les contours détectés dans la première image. Une recherche exhaustive de la pose dans l'espace 3D est alors nécessaire (voir la figure 3.6).

3.3.2 Reconstruction 3D initiale contrainte au modèle

Une fois que la première pose de la caméra est obtenue par l'une des méthodes décrites précédemment, la méthode de reconstruction 3D par rétro-projection utilisée dans Bleser *et al.* (2006); Sourimant *et al.* (2007); Vacchetti et Lepetit (2004) peut alors être appliquée. Une première reconstruction des points 3D est effectuée par rétro-projection, sur le modèle, des points 2D détectés dans la première image. Un ensemble de points 3D appartenant au modèle, est ainsi obtenu. La rétro-projection consiste à effectuer pour chaque point 2D un lancer de rayon (faisceau passant par le centre optique de la caméra et le point 2D) et à calculer l'intersection entre ce faisceau et la surface du modèle 3D. La précision de cette reconstruction initiale dépend à la fois de la précision de l'estimation de la pose de la caméra ainsi

FIGURE 3.6 – Initialisation du SLAM par recherche exhaustive des contours du modèle dans l'image : calcule de la première pose de la caméra. Fonctionnement typique de l'algorithme proposé par Prisacariu et Reid (2011) pour une image : (a) - initialisation, (b) - itération de descente de gradient, (c) - résultat final.

que de la précision du modèle.

3.4 Conclusion

Dans ce chapitre nous présentons le principe général ainsi que le formalisme proposé pour le SLAM contraint. Comme nous le montrerons par la suite, ce cadre de résolution fait de cette solution un outil efficace et générique pour la localisation d'une caméra en environnement partiellement connu. Cette généricité est caractérisée par une capacité à gérer :

- ▷ plusieurs types d'objets (texturés/peu texturés, plans ou 3D) et des primitives de différentes natures (points, segments, *etc.*) ;
- ▷ plusieurs types de modèles (modèle géométrique, modèle photo-géométrique composé d'un nuage de points 3D, modèle surfacique) ;
- ▷ la combinaison de différentes contraintes.

CHAPITRE

4

Unification SLAM et localisation par rapport à un modèle 3D connu

Dans ce chapitre, nous présentons l'approche unifiant une localisation de type SLAM avec celle de type suivi basé modèle, introduite dans le chapitre précédent. Dans les sections 4.2 et 4.3 sera respectivement introduit l'ajustement de faisceaux contraint, par des segments 3D et par des points 3D. Les avantages de ces derniers sont démontrés sur des données de synthèse et réelles. Une partie de ces travaux a été publiée dans Tamaazousti et al. (2011c).

4.1 Introduction

Dans ce chapitre, nous considérons le cas où nous disposons d'un modèle 3D de l'objet d'intérêt qui est relativement précis et où ce modèle dispose de primitives pouvant être mises en correspondance avec les primitives image. Nous proposons ici une solution permettant d'unifier les approches de type SLAM et de suivi basé modèle. Le principe est donc d'initialiser la carte 3D du SLAM avec les primitives du modèle. Ces primitives, une fois associées avec les primitives image leur correspondant, induisent une contrainte sur la localisation. La localisation optimale correspond alors à celle qui minimise l'erreur de reprojection des primitives pré-reconstruites (modèle 3D) ainsi que l'erreur de reprojection des primitives reconstruites en ligne (SLAM). Ceci est obtenu à l'aide d'un ajustement de faisceaux contraint par la carte 3D pré-reconstruite inhérente aux méthodes de suivi basé modèle. En effet, étant donné que ces primitives extraites du modèle sont considérées parfaites, leurs positions ne sont pas optimisées dans le processus de SLAM (c'est-à-dire qu'elles ont 0-DoF dans le processus d'ajustement de faisceaux). Néanmoins, leurs erreurs de reprojection sont prises en compte dans l'ajustement de faisceaux.

Elles permettent alors de contraindre directement le mouvement de la caméra et donc, indirectement, les primitives reconstruites en ligne par le processus de SLAM.

Afin de souligner les avantages de notre cadre de SLAM contraint, deux situations suivantes sont considérées.

- ▷ L'objet est visible dans l'image : les primitives qui ont 0-DoF fournissent le recalage avec le repère initial, le facteur d'échelle et évitent l'accumulation d'erreurs. Par conséquent, puisque la localisation de la caméra est améliorée, les primitives reconstruites par le SLAM sont plus précises.
- ▷ L'objet est occulté ou prend une faible partie dans l'image : les primitives 3D reconstruites en ligne par le processus de SLAM permettent de garder un suivi précis aussi longtemps que l'objet est statique par rapport à son environnement.

Notons que les primitives reconstruites en ligne par le SLAM (points 3D) peuvent être d'une nature différente de celle issues du modèle (points 3D ou segments 3D).

4.2 Ajustement de faisceaux contraint par des segments 3D

Lorsque l'objet d'intérêt est peu texturé et qu'un modèle géométrique de cet objet est disponible, l'unification du SLAM avec une méthode de suivi basé contour est un choix intéressant. Afin de pouvoir gérer des objets 3D peu texturés, une contrainte avec des segments 3D extraits du modèle est proposée dans cette section.

La carte initiale des segments 3D est obtenue, hors ligne, en échantillonnant les arêtes franches du modèle géomètrique de l'objet d'intérêt. Pendant le processus de SLAM, ces segments 3D sont projetés dans les images et appariés aux contours les plus proches qui ont une orientation similaire. Les distances entre les projections des points milieux des segments 3D dans les images et leurs points de contour correspondants (distance contour au modèle) sont alors des distances supplémentaires à minimiser au cours du processus d'optimisation. En particulier, il s'agit de la distance orthogonale au segment 3D projeté dans l'image. Il en résulte un ajustement de faisceaux contraint qui minimise une fonction de coût à deux termes : le premier terme correspond à la somme des erreurs de reprojection entre les points reconstruits et leurs observations (1.51), le second correspond à la somme des distances contour au modèle (4.1). La stratégie de pondération de ces deux termes est détaillée en section 3.2.5 du chapitre 3. Le tableau 4.1 présente en bleu le nombre de degrés de liberté attribué aux différentes primitives : 3 pour les points reconstruits par le SLAM et 0 pour les segments extraits du modèle.

A noter que, les segments ont précédemment été utilisés par Klein et Murray (2008) pour améliorer la robustesse du SLAM aux mouvements de caméra rapide. Ils proposent alors un processus complet, incluant l'extraction de contours et appa-

Primitives reconstruites pSLAM ($\mathcal{E}_E$)	Primitives pré-reconstruites Modèle 3D ($\mathcal{E}_M$)	
Points 3D	Points 3D	Segments 3D
3-DoF	0-DoF	**0-DoF**

TABLE 4.1 – **Unification SLAM et localisation par rapport à un modèle 3D géométrique, dans le cas du pSLAM.** Ajustement de faisceaux contraint par des segments 3D extraits du modèle géométrique. Les contraintes sont exprimées en fonction du nombre de degrés de liberté (DoF) attribués aux différentes primitives : 3 pour les points reconstruits par le pSLAM et 0 pour les segments extraits du modèle.

riement dans les images clés successives, triangulation des segments et un ajustement de faisceaux dédié à combiner les points et les segments. Dans ce chapitre, nous considérons que nous possédons déjà une carte de segments 3D fournie par le modèle géométrique de l'objet d'intérêt. Nous utilisons cette information disponible pour améliorer la précision d'un algorithme pSLAM basé images clés.

4.2.1 Contrainte aux segments 3D extraits du modèle

La contrainte présentée ici utilise des segments extraits du modèle 3D pour contraindre implicitement les poses de la caméra. La contrainte basée modèle est alors fournie par les arêtes franches des contours englobants du modèle 3D géométrique : pour un modèle sous forme de maillage ou en facettes (un ensemble de triangles), les arêtes formées par deux triangles et dont l'angle diédral est supérieur à un certain seuil. Ces arêtes franches sont ensuite échantillonnées en un ensemble de petits segments 3D $\{\mathbf{L}_i\}_{i=1}^{l}$; chacun étant paramétrisé par un point milieu $\mathbf{M}_i$ et un vecteur directeur $\mathbf{D}_i$. Ces segments constituent, la partie connue de l'environnement. A noter que d'autres méthodes d'extraction de segments peuvent être envisagées comme la méthode d'extraction de contour à partir d'un modèle 3D proposée par Platonov et Langer (2007).

4.2.2 Fonction de coût

La fonction de coût, associée à la contrainte basée sur les segments 3D, minimise la distance orthogonale entre la projection $\mathsf{P}_j\mathbf{M}_i$ du point milieu $\mathbf{M}_i$ du segment 3D et le point de contour $\mathbf{m}_{i,j}$ auquel il est associé dans l'image (voir par exemple Wuest *et al.* (2007a) pour plus de détails). Rappelons que P_j est la matrice de projection associée à la caméra $\mathcal{C}_j$. Considérons des associations 3D-2D entre des segments 3D $\mathbf{L}_i$ du modèle et des contours $\mathbf{m}_{i,j}$ extraits aux images clés $j \in \mathcal{S}_i$ (voir la section 4.2.3), la contrainte induite par des segments est donnée par l'équation (4.1).

$$\mathcal{E}_M\left(\{\mathsf{R}_j,\mathbf{t}_j\}_{j=1}^{m}\right)=\sum_{i=1}^{l}\sum_{j\in\mathcal{S}_i}|\mathbf{n}_{i,j}.\left(\mathbf{m}_{i,j}-\mathsf{P}_j\mathbf{M}_i\right)|, \tag{4.1}$$

où $\mathbf{n}_{i,j}$ est la normale à la projection dans l'image $\mathsf{P}_j\mathbf{D}_i$ du vecteur directeur $\mathbf{D}_i$ du segment 3D $\mathbf{L}_i$. A noter que cette contrainte est exprimée en pixels ce qui facilite la pondération des différents termes de la fonction de coût totale (voir la section 3.2.5). Comparativement aux méthodes de l'état de l'art de suivi basé contour (Drummond et Cipolla (2002); Wuest *et al.* (2007a)), cette fonction de coût utilise sensiblement le même critère mais étendu au cas multi-vues : il s'applique simultanément sur l'ensemble des images traitées dans l'ajustement de faisceaux.

4.2.3 Associations 3D-2D entre les segments 3D et les contours image

Cette étape d'association 3D-2D est réalisée uniquement aux images clés pour le processus d'ajustement de faisceaux contraint. Pour l'équation (4.1), les segments 3D extraits du modèle doivent être associés aux contours dans l'image. Un test de visibilité est utilisé dans un premier temps pour ne traiter qu'un sous-ensemble de segments 3D visibles pour chaque image clé de l'ajustement de faisceaux. Les segments 3D visibles sont alors projetés dans chacune d'elles, puis une recherche est réalisée, le long de la normale au segment projeté, afin de trouver le maximum de gradient, dans un petit voisinage. En pratique nous observons que, choisir uniquement le contour le plus proche avec l'orientation la plus similaire est suffisant, puisque la pose initiale estimée par le SLAM est proche de la solution. Il n'y a donc pas besoin d'hypothèses multiples pour l'association 3D-2D.

Les étapes de l'ajustement de faisceaux contraint par segments sont récapitulées dans le tableau 4.2. La fonction de coût globale minimisée est décrite par l'équation (4.2).

$$\begin{aligned}\mathcal{E}\left(\{\mathsf{R}_j,\mathbf{t}_j\}_{j=1}^{m},\{\mathbf{Q}_i\}_{i=1}^{N}\right) &= \underbrace{\sum_{i=1}^{N}\sum_{j\in\mathcal{A}_i}\rho\left(d^2(\mathbf{q}_{i,j},\mathsf{P}_j\mathbf{Q}_i),c_1\right)}_{\text{Partie inconnue de l'environnement }(\mathcal{E}_E)}\\ &\quad+\underbrace{\sum_{i=1}^{l}\sum_{j\in\mathcal{S}_i}\rho\left(|\mathbf{n}_{i,j}.\left(\mathbf{m}_{i,j}-\mathsf{P}_j\mathbf{M}_i\right)|,c_2\right)}_{\text{Partie associée au modèle 3D }(\mathcal{E}_M)}.\end{aligned} \tag{4.2}$$

▷ Test de visibilité pour trouver le sous-ensemble de segments 3D visibles pour chaque image clé $j \in \mathcal{S}_i$.
▷ Projection des segments 3D visibles $\mathbf{L}_i$ dans chaque image clé $j \in \mathcal{S}_i$.
▷ Association des segments 3D $\mathbf{L}_i$ aux contours de l'image $\mathbf{m}_{i,j}$.
▷ Calcul des seuils de rejet c_1 et c_2.
▷ Minimisation de l'équation (4.2) avec l'algorithme de Levenberg-Marquardt.

TABLE 4.2 – Etapes de l'ajustement de faisceaux contraint par des segments 3D extraits du modèle.

4.3 Ajustement de faisceaux contraint par des points 3D

Nous pouvons proposer le même principe d'approche lorsque le modèle 3D n'est pas un modèle filaire mais un nuage de points 3D avec une signature photométrique [1].

Dans ce cas, il s'agit de l'unification du SLAM avec les méthodes de localisation par rapport à un nuage de points 3D connus. Dans cette section une nouvelle contrainte est alors décrite. L'idée générale est d'unifier deux types de points, ceux pré-reconstruits hors ligne (le modèle 3D) supposés parfait et ceux reconstruits en ligne par le SLAM dont les positions doivent être optimisées. A noter que les positions des points 3D préalablement reconstruits sont considérées parfaites et ne sont donc pas remises en question. Cela se traduit dans l'ajustement de faisceaux contraint par le fait que ces points possèdent zéro degré de liberté, comme dans le cas précédent avec les segments. Ces points 3D pré-reconstruits sont similaires à ceux issus de la reconstruction du SLAM, excepté que leurs positions sont considérées constantes (0-DoF points). Il en résulte alors un ajustement de faisceaux contraint qui minimise une fonction de coût à deux termes : le premier correspond à l'erreur de reprojection des points (3-DoF) reconstruits par le SLAM alors que le second correspond à l'erreur de reprojection des points (0-DoF) fournis par le nuage de points pré-reconstruit. Le tableau 4.3 présente en bleu le nombre de degrés de liberté attribué aux différentes primitives.

A noter que cette contrainte permet de gérer principalement les objets texturés. En effet, il faut au préalable pouvoir reconstruire un nuage de points 3D de l'objet ce qui nécessite la présence de texture sur celui-ci.

1. Ce nuage de point peut être issu d'une phase préalable d'apprentissage pouvant être réalisée par exemple avec une méthode de type SfM (voir la section 2.1.1).

Primitives reconstruites pSLAM ($\mathcal{E}_E$)	Primitives pré-reconstruites Modèle 3D ($\mathcal{E}_M$)	
Points 3D	Points 3D	Segments 3D
3-DoF	**0-DoF**	0-DoF

TABLE 4.3 – **Unification SLAM et localisation par rapport à un modèle 3D photo-géométrique, dans le cas du pSLAM.** Ajustement de faisceaux contraint par un nuage de points 3D connu. Les contraintes sont exprimées en fonction du nombre de degrés de liberté (DoF) attribués aux différentes primitives : 3 pour les points reconstruits par le pSLAM et 0 pour les points pré-reconstruits.

4.3.1 Contrainte par un nuage de points 3D pré-existant

La contrainte présentée ici utilise un nuage de points 3D $\left\{\hat{\mathbf{Q}}_i\right\}_{i=1}^{p}$ pour lesquels un ou plusieurs descripteurs sont associés (information photométrique). Un tel nuage de points est habituellement obtenu par un processus préalable de reconstruction multi-vues de type SfM ou directement extrait à partir de la texture d'un modèle CAO. Ce nuage de points, constituant la partie connue de l'environnement, est utilisé pour contraindre implicitement la trajectoire de la caméra estimée par le processus SLAM, en y intégrant les résidus associés dans l'ajustement de faisceaux (voir le tableau 4.3).

4.3.2 Fonction de coût

La fonction de coût, associée à la contrainte induite par des points 3D, minimise la distance euclidienne entre la projection dans l'image du point 3D pré-reconstruit $\mathsf{P}_j\hat{\mathbf{Q}}_i$, et le point d'intérêt 2D $\hat{\mathbf{q}}_{i,j}$ auquel il est associé. Considérons des associations 3D-2D entre des points 3D $\hat{\mathbf{Q}}_i$ du modèle et des points d'intérêt $\hat{\mathbf{q}}_{i,j}$ extraits aux images clés $j \in \mathcal{P}_i$ (voir la section 4.3.3), la contrainte par des points est donnée par l'équation suivante :

$$\mathcal{E}_M\left(\{\mathsf{R}_j, \mathbf{t}_j\}_{j=1}^{m}\right) = \sum_{i=1}^{p} \sum_{j \in \mathcal{P}_i} d^2(\hat{\mathbf{q}}_{i,j}, \mathsf{P}_j\hat{\mathbf{Q}}_i). \tag{4.3}$$

Cette fonction de coût est exprimée en pixels ce qui facilite la pondération des différents termes de la fonction de coût totale (voir la section 3.2.5). Comparativement aux méthodes de suivi basé points du modèle (Nistér et Stewenius (2006)), cette fonction de coût utilise le même critère mais étendu au cas multi-vues : il s'agit de l'erreur de reprojection sur l'ensemble des images traitées dans l'ajustement de faisceaux.

4.3.3 Associations 3D-2D entre les points 3D et les points d'intérêt image

Cette étape d'association 3D-2D est réalisée uniquement aux images clés pour le processus d'ajustement de faisceaux contraint.

Pour l'équation (4.3), les points 3D qui composent le modèle sont associés aux points d'intérêt dans l'image. Ces points 3D sont projetés dans chaque image clé, puis une recherche est réalisée, dans le voisinage du point projeté, afin de trouver le point d'intérêt le plus semblables. En pratique, la zone de recherche dans laquelle s'effectue la mise en correspondance est petite [2] car la pose estimée par le SLAM est proche de la solution. Cela implique peu de comparaisons de descripteurs et donc un faible coût au niveau du temps de traitement. Il s'agit donc d'une étape de mise en correspondance classique de points lors de laquelle nous conservons l'association qui fournit les points les plus semblables.

A noter que si un modèle géométrique surfacique est disponible en plus du nuage de points 3D, il est alors possible de réaliser un test de visibilité dans un premier temps pour ne traiter qu'un sous ensemble de points 3D visibles pour chaque image clé inclue dans l'ajustement de faisceaux.

Les étapes de l'ajustement de faisceaux contraint par points 3D sont récapitulées dans le tableau 4.4. La fonction de coût minimisée est décrite par l'équation (4.4).

$$\mathcal{E}\left(\{\mathsf{R}_j, \mathbf{t}_j\}_{j=1}^{m}, \{\mathbf{Q}_i\}_{i=1}^{N}\right) = \underbrace{\sum_{i=1}^{N}\sum_{j\in\mathcal{A}_i}\rho\left(d^2(\mathbf{q}_{i,j}, \mathsf{P}_j\mathbf{Q}_i), c_1\right)}_{\text{Partie inconnue de l'environnement } (\mathcal{E}_E)} + \underbrace{\sum_{i=1}^{p}\sum_{j\in\mathcal{P}_i}\rho\left(d^2(\hat{\mathbf{q}}_{i,j}, \mathsf{P}_j\hat{\mathbf{Q}}_i), c_2\right)}_{\text{Partie associée au modèle 3D } (\mathcal{E}_M)} \tag{4.4}$$

▷ Projection des points 3D visibles $\hat{\mathbf{Q}}_i$ dans chaque image clé $j \in \mathcal{P}_i$.
▷ Association des points 3D $\hat{\mathbf{Q}}_i$ aux points d'intérêt de l'image $\hat{\mathbf{q}}_{i,j}$.
▷ Calcul des seuils de rejet c_1 et c_2.
▷ Minimisation l'équation (4.4) avec l'algorithme de Levenberg-Marquardt.

TABLE 4.4 – Etapes de l'ajustement de faisceaux contraint par un nuage de points 3D.

2. Une zone de 11×11 pixels centrée sur la projection dans l'image du point 3D.

4.4 Structure creuse de l'ajustement de faisceaux contraint au modèle

Cette section, montre que la structure creuse des matrices Hessienne et Jacobienne, associées à l'ajustement de faisceaux, est conservée malgré l'ajout de la contrainte par segments ou points 3D.

La figure 4.1 représente un exemple d'illustration d'ajustement de faisceaux contraint par segments ainsi que les matrices Hessienne et Jacobienne mises en jeux. Une propriété importante est que ces matrices conservent une structure par blocs. En effet, des résidus dépendant uniquement des paramètres de la caméra sont ajoutés dans la matrice Jacobienne (un résidu pour chaque observation d'une primitive 3D). De plus, le nombre de paramètres n'augmente pas puisque les primitives 3D de la carte pré-reconstruite ne sont pas optimisées. La conséquence est que la matrice Hessienne possède exactement la même structure que pour un ajustement de faisceaux classique.

Ainsi, il est possible d'implémenter de manière efficace l'ajustement de faisceaux contraint, en prenant en compte cette structure par blocs, comme décrit dans Triggs *et al.* (2000).

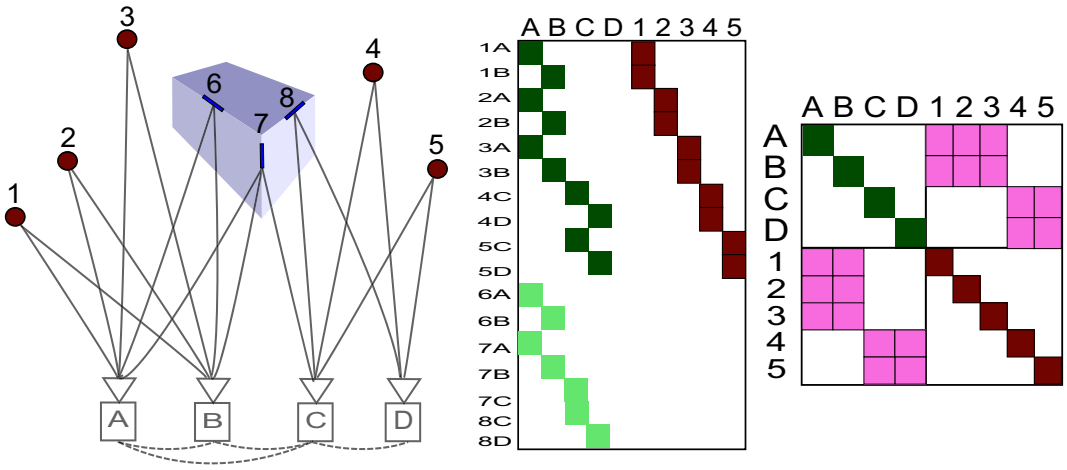


FIGURE 4.1 – A gauche : un exemple d'ajustement de faisceaux contraint par des segments 3D extraits du modèle. En rouge, les points 3D de la partie inconnue de l'environnement (de 1 à 5) ; en bleu, les segments 3D associés à la partie connue de l'environnement (de 6 à 8). Les carrés (de A à D) représentent la trajectoire de la caméra. Les lignes noires indiquent l'observabilité des primitives 3D pour chaque pose de la caméra en mouvement. A droite : les matrices Jacobienne et Hessienne associées.

4.5 Résultats expérimentaux

Cette section évalue la méthode de localisation unifiant le SLAM et le suivi basé modèle. L'algorithme de SLAM basé images clés, tel que décrit dans Mouragnon *et al.* (2006), est utilisé dans les différentes expérimentations. Il fonctionne en temps réel grâce à un ajustement de faisceaux local appliqué sur une fenêtre glissante de triplet d'images clés. A chaque image clé, seules les trois dernières poses de la caméra (associées aux trois dernières images clés) ainsi que les points 3D qu'elles observent, sont optimisés (voir la section 1.4).

A noter que, l'étape d'initialisation de Mouragnon *et al.* (2006), a été modifiée pour permettre de recaler le repère de la reconstruction du SLAM avec celui du modèle 3D de l'objet (voir la section 3.3). Dans sa version initiale, elle est réalisée par l'algorithme des cinq points, sur les trois premières images clés. Elle est ici réalisée de la manière suivante pour les contraintes segments et points :

- ▷ Contrainte segments : nous supposons dans ce cas qu'une estimation approximative de la première pose de la caméra est disponible. Cette estimation est ensuite améliorée par un algorithme de raffinement sur les contours de l'objet dans la première image. Un nuage de points 3D initial est ensuite obtenu en faisant une rétro-projection des observations de la première image.
- ▷ Contrainte points : dans ce cas, la différence est que nous disposons déjà d'un nuage de points que nous utilisons directement pour initialiser le SLAM. La première pose est estimée par mise en correspondance 3D-2D (voir la section 1.3.4).

4.5.1 Contrainte segments

Dans cette section, des évaluations de l'ajustement de faisceaux contraint par des segments 3D sont présentées sur des données de synthèse et réelles.

Une évaluation des performances, pour le suivi d'objet 3D, de l'algorithme de SLAM, est réalisée sur une séquence de synthèse, avec deux types d'ajustement de faisceaux local :

- ▷ Celui décrit dans Mouragnon *et al.* (2006) dans sa version originale appelé par la suite LBA_E [3]. Il minimise l'équation (1.51) par l'algorithme de Levenberg-Marquardt.
- ▷ L'algorithme LBA_CS&E [4] proposé ici. Il minimise l'équation (4.2) avec la procédure décrite dans le tableau 4.2.

Une comparaison est ensuite effectuée, sur des données réelles entre l'algorithme de SLAM avec l'ajustement de faisceaux contraint (SLAM + LBA_CS&E) et l'état de l'art en suivi d'objet 3D (Vacchetti *et al.* (2004); Wuest *et al.* (2005)).

3. E désigne le fait que les relations multi-vues associées aux primitives de la partie inconnue de l'environnement sont prises en compte.

4. CS désigne le fait que la contrainte par des segments 3D est prise en compte dans l'ajustement de faisceaux.

4.5.1.1 Evaluation sur des données de synthèse

Cette section détaille l'expérimentation réalisée pour comparer les deux algorithmes d'ajustement de faisceaux LBA_E et LBA_CS&E, sur une séquence de synthèse. Après avoir décrit brièvement la séquence d'étude, l'apport de l'algorithme LBA_CS&E dans un SLAM est évalué, en terme de précision de localisation. La qualité de la localisation est mesurée par les erreurs 3D en position et en orientation entre la vérité terrain et la pose estimée après l'ajustement de faisceaux local à chaque image clé.

La séquence « double cubes ». Dans la séquence « double cubes », illustrée par la figure 4.2, la scène est composée de deux cubes situés sur un sol texturé avec d'autres petits cubes qui les occultent partiellement. L'objet d'intérêt, c'est-à-dire pour lequel un modèle 3D est disponible, est composé des deux cubes principaux. La trajectoire de la caméra est un cercle dont le rayon est de trois mètres autour des principaux cubes. La hauteur cumulée des deux cubes superposés est d'un mètre et demi.

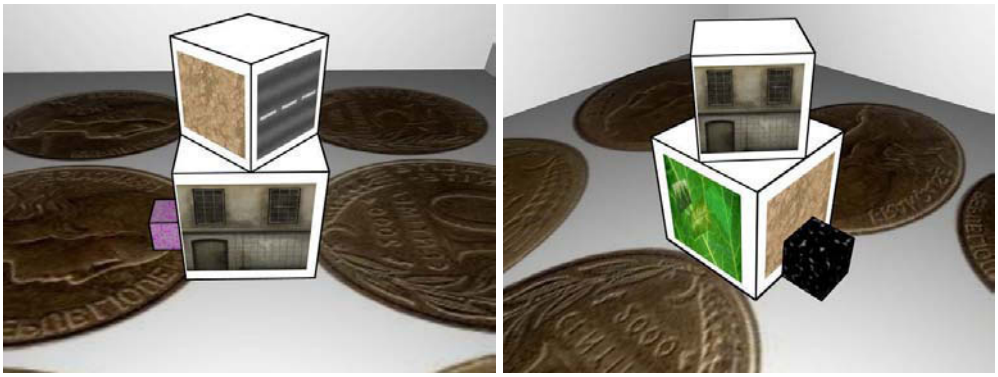


FIGURE 4.2 – Deux images de la séquence « double cubes ».

Choix de la combinaison. Nous comparons sur cette séquence, les trois combinaisons proposées dans la section 3.2.5 pour l'algorithme LBA_LC&E, pour calculer le poids entre les deux termes de la fonction de coût. Pour rappel nous évaluons les trois combinaisons suivantes :

- ▷ combinaison 1 : $c_1 = c_{Env}$ et $c_2 = c_{Modele}$
- ▷ combinaison 2 : $c_1 = c_2 = c_{All}$
- ▷ combinaison 3 : $c_1 = c_2 = c_{Max} = \max(c_{Env}, c_{Modele})$

Notons que la pose initiale de la caméra associée à la première image est donnée par la vérité terrain. Les résultats de la localisation sont représentés sur la figure 4.4. La combinaison 2 présente le plus mauvais résultat en terme de précision. Cela peut s'expliquer par le fait que le seuil de rejet de l'estimateur robuste est estimé pour une distribution des résidus unimodale. Il apparait sur la figure 4.4 (a) que

cette hypothèse n'est pas vérifiée en pratique. Comme pressenti, l'ordre de grandeur des erreurs résiduelles associées au modèle est généralement supérieur à celui de l'environnement. Ainsi cette combinaison entraine un seuil de rejet sous-estimé qui élimine trop de résidus associés au modèle lors de l'optimisation. Finalement, la combinaison 3 donne de meilleurs résultats que la 1 et la 2. Ce constat a été vérifié sur plusieurs séquences de synthèse. Cela montre que les primitives contraintes au modèle doivent être favorisées durant le processus d'optimisation, tout en vérifiant les relations multi-vues de la partie inconnue de l'environnement. Dans la suite, nous ne considèrerons plus que la combinaison 3 pour l'équation (4.2).

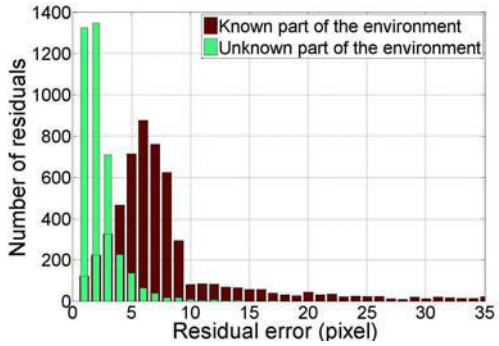
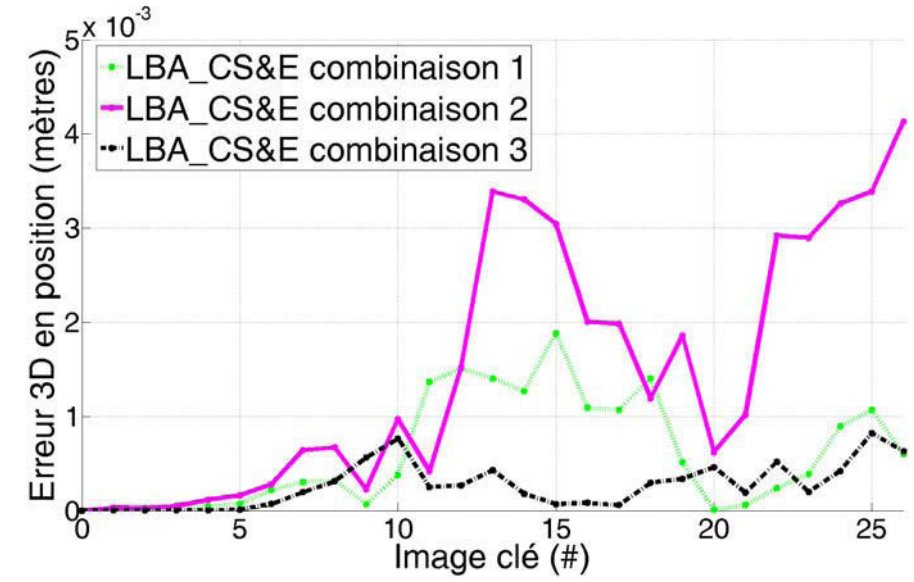


FIGURE 4.3 – Distribution des erreurs résiduelles. En vert, (respectivement en rouge) la distribution des erreurs résiduelles associées à la partie inconnue (respectivement la partie connue) de l'environnement.

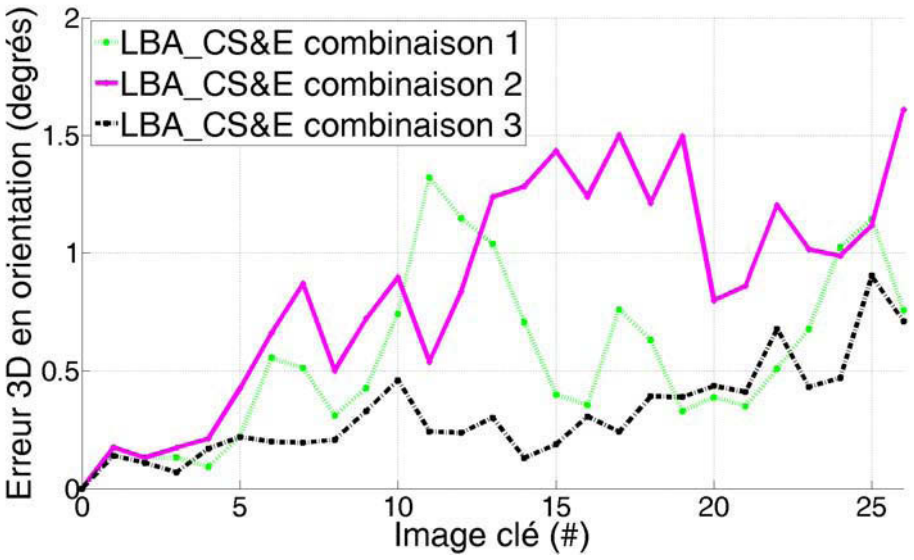
Comparaison des deux algorithmes LBA. La première expérimentation a été réalisée sur cette séquence pour comparer les algorithmes de raffinement LBA_E et LBA_CS&E. Elle évalue la précision de localisation au cours de la séquence avec une initialisation parfaite donnée par la vérité terrain. Les résultats sont présentés sur les figures 4.5(a) (erreurs 3D en position) et 4.5(b) (erreurs 3D en orientation). Le SLAM avec l'ajustement de faisceaux local classique LBA_E est sujet à des accumulations d'erreurs alors qu'avec l'ajustement de faisceaux local contraint LBA_CS&E, la dérive est fortement réduite voire supprimée. Par exemple, avec le LBA_E l'erreur en position est de 3 centimètres à l'image clé 10 et elle augmente jusqu'à 30 centimètres pour l'image clé 26. Alors qu'avec le LBA_CS&E, la variation de l'erreur est très faible durant toute la séquence. L'ajout de la contrainte améliore de manière significative la précision de la localisation.

La seconde expérimentation évalue la robustesse du SLAM avec l'algorithme LBA_CS&E à une initialisation approximative. Des perturbations de plus en plus élevées sur la position [5], sont appliquées à la pose initiale de la caméra, correspondant à la première image. Leurs amplitudes varient de 1% à 6% de la longueur du

5. Les orientations des poses de la caméra ne sont pas perturbées.



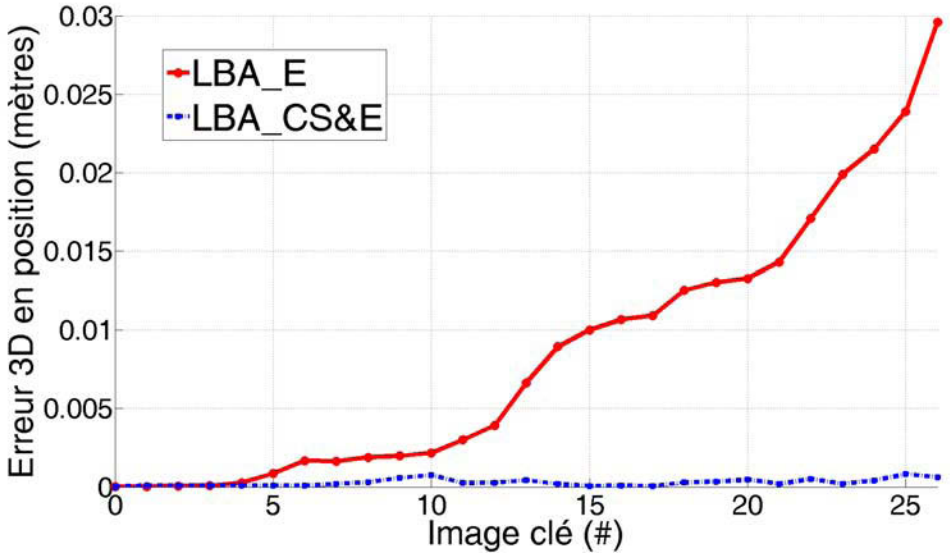
(a)



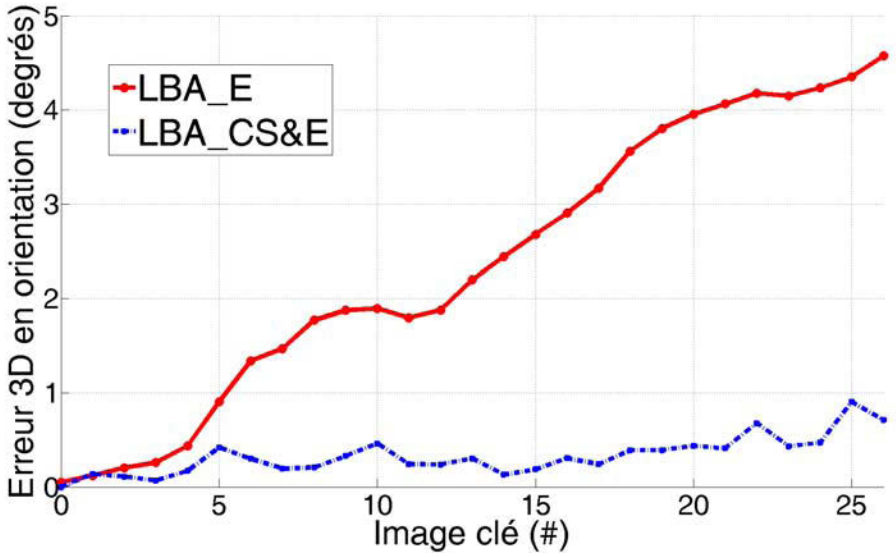
(b)

FIGURE 4.4 – Evaluation de la stratégie de pondération proposée. Erreurs en position (a) et en orientation (b) des poses de la caméra, pour les différentes combinaisons.

rayon du cercle formé par la trajectoire de la caméra (un exemple de perturbation de 6% est illustré sur la figure 4.6(a)). Les résultats sont une moyenne sur dix tirages dans des directions aléatoires. La figure 4.6(c) montre que le SLAM avec l'ajustement de faisceaux local contraint par segments gère des erreurs d'initialisation : après quelques poses de la caméra, les erreurs 3D sont stabilisées à de faibles valeurs. Par exemple, pour une amplitude de 6% qui correspond à une erreur en posi-



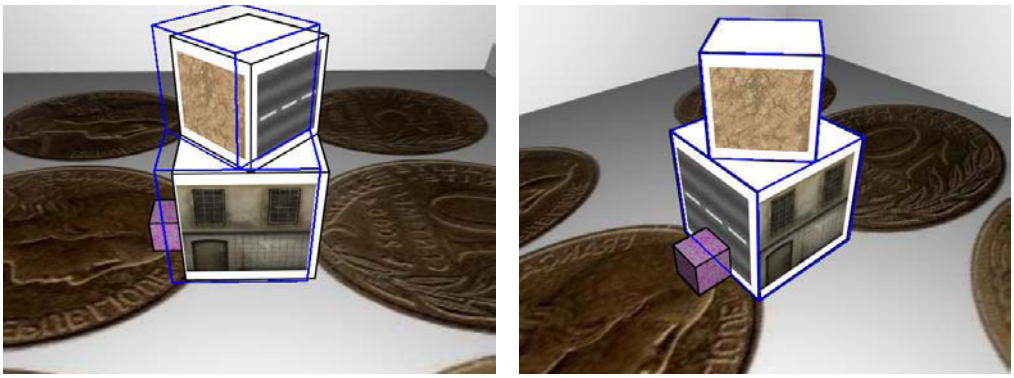
(a)



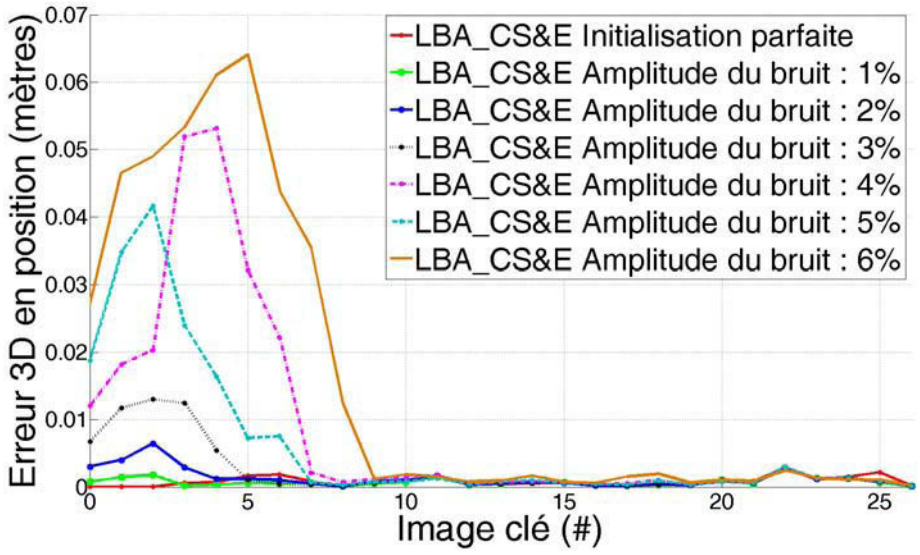
(b)

FIGURE 4.5 – Evaluation de la précision de la localisation du SLAM contraint par segments 3D et comparaison avec le SLAM classique. (a) et (b) représentent les erreurs en position et en orientation.

tion de 30 centimètres, l'erreur après quelques images clés descend à 3 centimètres. Notons que le résultat concernant la robustesse à l'initialisation pour l'algorithme LBA_E, n'est pas présenté ici car il est évident qu'il ne peut pas corriger cette erreur d'initialisation.



(a) image 1 (b) image 42



(c)

FIGURE 4.6 – Evaluation de la robustesse à une mauvaise initialisation du SLAM contraint par des segments 3D. (c) Les résultats obtenus par le SLAM avec l'ajustement de faisceaux contraint par segments (LBA_CS&E) pour différentes amplitudes de perturbation ($\in [1\% \cdots 6\%]$ du rayon de la trajectoire de la caméra) appliquées sur la première pose de la caméra. (a) Illustration du décalage dans l'image entre la reprojection du modèle et l'objet d'intérêt, pour une perturbation de 6%.

4.5.1.2 Evaluation sur des données réelles

Dans cette section, le SLAM avec l'ajustement de faisceaux contraint LBA_CS&E est comparé à l'état de l'art des méthodes de suivi basé modèle sur des objets peu texturés. Pour cela, des séquences ont été réalisées avec une caméra IEEE1394 GUPPY, fournissant des images (640×480) à une fréquence de 30 images par seconde.

Suivi d'objet 3D peu texturé.

L'objet d'intérêt est une maquette miniature représentant la Lamborghini Gallardo. Le modèle 3D géométrique utilisé dans les expérimentations est composé d'environ 14000 triangles. 1186 segments 3D, sont alors extraits de celui-ci, comme illustré sur la figure 4.7. La voiture est placée sur un bureau composé d'un écran et d'un clavier d'ordinateur, de livres, *etc*. Ils constituent la partie inconnue de l'environnement. Une comparaison est effectuée entre le SLAM avec LBA_CS&E et un algorithme de suivi basé modèle géométrique, qui par définition utilise uniquement la partie connue de l'environnement. Ce dernier est une version améliorée de l'algorithme de suivi basé modèle proposé par Drummond et Cipolla (2002) qui inclut, en plus, une notion d'hypothèses multiples pour l'association de données, comme décrit dans Vacchetti *et al.* (2004); Wuest *et al.* (2005). La comparaison est réalisée sur une séquence difficile qui présente des variations d'échelle importantes, des mouvements rapides, des occultations partielles de la voiture, *etc*. A noter que comme l'objet d'intérêt est peu texturé, l'initialisation (c'est-à-dire le recalage sur la première image) est obtenue en positionnant approximativement, pour la première image, un marqueur codé près de la voiture (voir la section 3.3.1).

FIGURE 4.7 – Les segments 3D extraits du modèle de la Lamborghini.

Résultats. La figure 4.8 présente les résultats obtenus par le SLAM avec LBA_CS&E et par l'algorithme de suivi basé modèle, sur cette séquence. Pour l'initialisation, le marqueur codé étant positionné approximativement prés de la voiture, le recalage initial sur la première image n'est donc pas précis. Les deux méthodes corrigent cette erreur après quelques images : l'avant et l'arrière de la voiture se projettent correctement sur les images, comme présenté sur la figure 4.8 à gauche.

Cependant, le SLAM avec LBA_CS&E fournit de meilleurs résultats que l'algorithme de suivi basé modèle. En effet, ce dernier est mis en échec lors de mouvements rapides de la caméra (voir la figure 4.8 à droite). Ceci est dû à de mauvaises mises en correspondance 3D-2D. De plus, la localisation obtenue est instable, ce qui se traduit par des tremblements du modèle reprojeté dans les images (*jittering*). L'algorithme de localisation que nous proposons utilise simultanément les informations de la partie connue de l'environnement (modèle) et de la partie inconnue de l'environnement. Il en résulte une localisation précise, stable et robuste. Nous verrons dans le chapitre 6 que l'algorithme SLAM avec LBA_CS&E est donc parfaitement adapté pour des applications de réalité augmentée en temps réel.

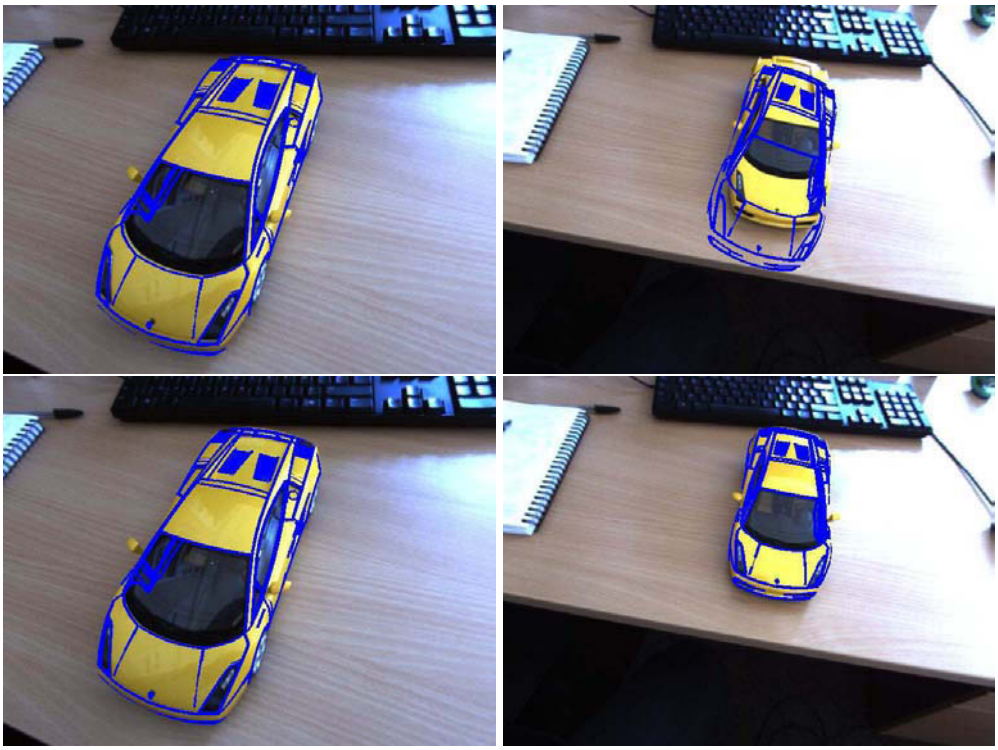


FIGURE 4.8 – Localisation dans un environnement partiellement connu, composé d'un objet 3D non texturé. En haut : les résultats obtenus avec un suivi basé modèle géométrique semblable à celui proposé dans Drummond et Cipolla (2002). En bas : les résultats obtenus avec le SLAM séquentiel qui utilise l'algorithme de raffinement LBA_CS&E.

Comparaison des deux algorithmes LBA sur la séquence cuisine.

L'objet d'intérêt est une cuisine. Le modèle 3D géométrique utilisé dans les expérimentations est composé d'environ 30000 triangles. 2000 segments 3D, sont alors extraits de celui-ci, comme illustré sur la figure 4.7. Tout le reste de la pièce

constitue la partie inconnue de l'environnement (le sol, les prises, la vaisselle, la hotte). Une comparaison est effectuée entre le SLAM avec l'ajustement de faisceaux classique noté LBA_E et l'algorithme de raffinement proposé LBA_CS&E. La comparaison est réalisée sur une séquence difficile qui présente des variations d'éclairage importantes et des occultations partielles de la cuisine. A noter que comme l'objet d'intérêt est peu texturé, l'initialisation (c'est-à-dire le recalage sur la première image) est obtenue en positionnant approximativement, pour la première image, un marqueur codé sur le sol près de la cuisine.

Les résultats sont présentés sur la figure 4.9. Comme on a pu le constater sur les données de synthèse, le SLAM avec l'algorithme de raffinement LBA_E est sujet à des accumulations d'erreurs (figure 4.9 à gauche) alors qu'avec l'ajustement de faisceaux local contraint LBA_CS&E, il ne dérive pas (figure 4.9 à droite). L'ajout de la contrainte améliore de manière significative la localisation.

Robustesse à des imprécisions de modèle

Nous montrons ici que le méthode de SLAM contraint par des segments 3D est robuste à des imprécisions de modèle, notamment pour un modèle reconstruit par une méthode utilisant le capteur Kinect (Newcombe *et al.* (2011); Whelan *et al.* (2012)). La figure 4.10 montre que la méthode de SLAM contraint fonctionne aussi bien avec un modèle CAO (à gauche) qu'avec un modèle moins précis, par exemple obtenu par le capteur Kinect (à droite).

En effet, l'utilisation des points de l'environnement contribue à rendre la méthode naturellement robuste à des erreurs de modèle. De plus, l'utilisation d'un estimateur robuste permet également de rendre la méthode robuste à de mauvaises associations entre les segments 3D du modèle et les contours dans l'image. Cependant si le modèle est trop peu précis, la méthode ne compensera pas cette erreur. Il serait donc intéressant de voir à quel point la méthode est robuste à ces imprécisions de modèle. Notons tout de même qu'il serait difficile de mettre en place un protocole expérimental permettant de répondre avec certitude à cette question. En effet, cela est très dépendant de la géométrie de l'objet ainsi que de la scène observée.

4.5.2 Contrainte points

Dans cette section, des évaluations de l'ajustement de faisceaux contraint par un nuage de points 3D sont présentées sur des données de synthèse.

Une évaluation des performances pour le suivi d'objet 3D de l'algorithme de SLAM est réalisée sur une séquence de synthèse avec deux types d'ajustement de faisceaux local :

▷ Celui décrit dans Mouragnon *et al.* (2006) dans sa version originale appelé par la suite LBA_E [6]. Il minimise l'équation (1.51) par l'algorithme de

6. E désigne le fait que les relations multi-vues associées aux primitives de la partie inconnue de

FIGURE 4.9 – Comparaison des résultats de localisation obtenus avec le SLAM classique (à gauche) et le SLAM contraint par segments 3D (à droite).

Levenberg-Marquardt.

▷ L'algorithme LBA_CP&E [7] proposé ici. Il minimise l'équation (4.4) avec la procédure décrite dans le tableau 4.4.

Une comparaison est ensuite effectuée, sur des données réelles entre l'algorithme de SLAM avec l'ajustement de faisceaux contraint (SLAM + LBA_CP&E)

l'environnement sont prises en compte.

7. CP désigne le fait que la contrainte par des points 3D est prise en compte dans l'ajustement de faisceaux.

FIGURE 4.10 – **Robustesse à des imprécisions de modèle pour la méthode SLAM contraint par des segments 3D.** Résultats de localisation obtenus avec la méthode de SLAM contraint par des segments 3D du modèle, avec un modèle CAO (à gauche) et avec un modèle reconstruit par le capteur Kinect (à droite). Pour les deux modèles, 2000 segments 3D (représentés en rouge) ont été extraits et utilisés dans l'ajustement de faisceaux contraint par segments.

et l'état de l'art en suivi d'objet 3D.

4.5.2.1 Evaluation sur des données de synthèse

La séquence « double cubes texturés ». Dans cette séquence, illustrée par la figure 4.11, la scène est composée de deux cubes texturés situés sur un sol texturé

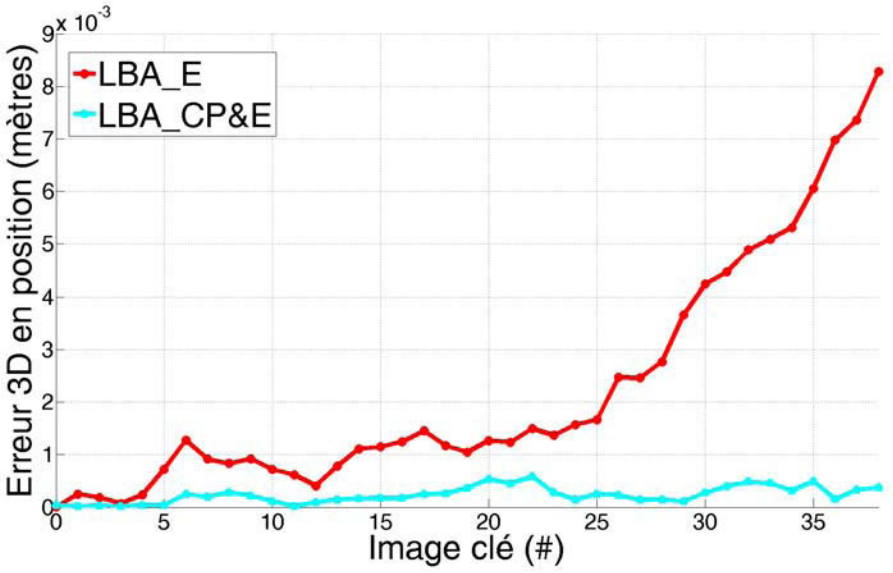
avec d'autres petits cubes qui les occultent partiellement. L'objet d'intérêt, c'est-à-dire pour lequel un nuage de points 3D constituant le modèle est disponible, est composé des deux cubes principaux. La trajectoire de la caméra est un cercle dont le rayon est de trois mètres autour des principaux cubes. La hauteur des deux cubes superposés est d'un mètre et demi.

FIGURE 4.11 – Deux images de la séquence « double cubes texturés ».

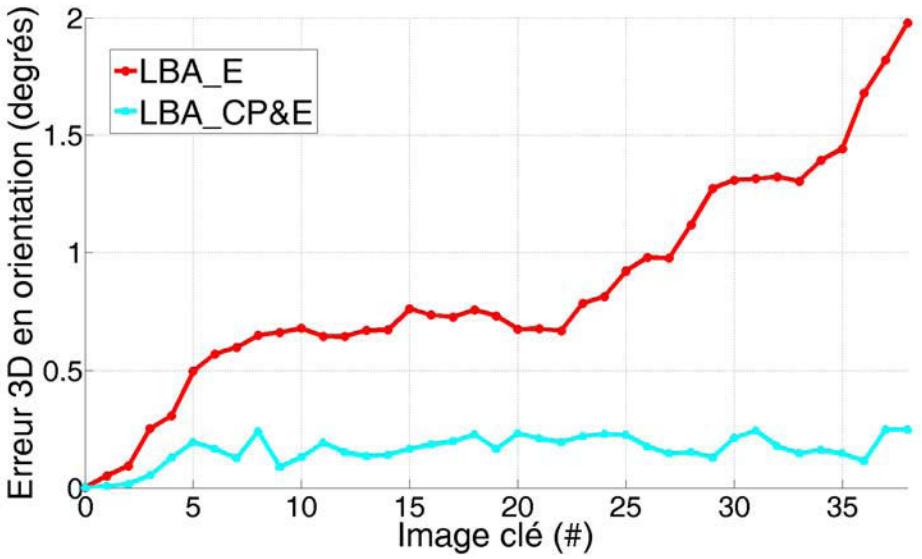
Comparaison des deux algorithmes LBA. Une expérimentation a été réalisée sur la séquence « double cubes texturés » pour comparer les algorithmes de raffinement LBA_E et LBA_CP&E. Elle évalue la précision de la localisation au cours de la séquence avec une initialisation parfaite donnée par la vérité terrain. Les résultats sont présentés sur la figure 4.12(a) (erreurs 3D en position) et 4.12(b) (erreurs 3D en orientation). Le SLAM avec l'algorithme de raffinement LBA_E est sujet à des accumulations d'erreurs alors qu'avec l'ajustement de faisceaux local contraint par points 3D LBA_CP&E, il ne dérive pas. Par exemple, avec le LBA_E l'erreur en orientation est de 0.2 degré pour les premières images clés et elle augmente jusqu'à 2 degrés pour l'image clé 38. Alors qu'avec le LBA_CP&E la variation de l'erreur est très faible durant toute la séquence, elle est inférieure à 0.2 degré. Comme pour les segments, l'ajout de la contrainte par points améliore de manière significative la précision de la localisation en évitant la dérive.

4.6 Conclusion

Ce chapitre présente l'approche de localisation unifiant un SLAM avec un suivi basé modèle. Deux processus d'ajustement de faisceaux contraint ont été introduits pour les algorithmes SLAM basés images clés. Le premier contraint l'ajustement de faisceaux par des segments 3D extraits d'un modèle géométrique (objets peu texturés) et le second par un nuage de points 3D (objets texturés). Deux fonctions de coût correspondant à ces contraintes ont été proposées. Ces fonctions prennent



(a)



(b)

FIGURE 4.12 – Evaluation de la précision de la localisation du SLAM contraint par points 3D et comparaison avec le SLAM classique. (a) et (b) représentent les erreurs en position et en orientation.

en compte les informations issues du modèle et les relations multi-vues du modèle et de l'environnement.

Des résultats expérimentaux, sur des données de synthèse, montrent que l'approche proposée fournit de meilleurs résultats que le SLAM « classique » en terme de précision et de robustesse à une mauvaise initialisation.

Une comparaison est ensuite effectuée sur des données réelles entre l'approche proposée SLAM avec ajustement de faisceaux contraint par segments (LBA_CS&E) et l'état de l'art en suivi d'objet 3D. Cette comparaison montre que l'approche proposée fournit de meilleurs résultats que les algorithmes de suivi basé modèle en terme de stabilité. En effet, la prise en compte de l'environnement permet de stabiliser la localisation et de conserver le suivi, même si l'objet d'intérêt est peu ou pas visible.

CHAPITRE

5

Unification SLAM et localisation avec mise à jour du modèle

Dans le chapitre précédent nous avons vu comment l'ajustement de faisceaux contraint permettait d'unifier le SLAM avec les approches de suivi basé modèle. Dans cette section, nous démontrons que l'ajustement de faisceaux contraint peut également permettre d'unifier le SLAM avec les approches de localisation par mise à jour du modèle. Pour cela nous introduisons la contrainte d'appartenance aux plans du modèle dans l'ajustement de faisceaux pour les primitives reconstruites par le SLAM. En section 5.2 nous nous intéressons à des primitives de type point (pSLAM) et en section 5.3 à des segments (sSLAM). Les résultats sur des données réelles et de synthèse démontrent l'apport de l'approche proposée. Une partie de ces travaux a été publiée dans Tamaazousti et al. (2011b).

5.1 Introduction

Dans le chapitre précédent nous avons présenté l'approche de localisation unifiant SLAM et suivi basé modèle. Nous avons vu que cette méthode fournit de bons résultats mais présente des limitations. En effet, dans le cas où le modèle est un nuage de points 3D, la méthode SLAM contraint au modèle utilise principalement l'apparence de l'objet pour la localisation. Cependant l'apparence de ce dernier peut évoluer au cours du temps et également en fonction des conditions d'éclairage. Les points 3D pré-reconstruits constituant le modèle, sont alors très difficiles à mettre en correspondance avec les points d'intérêt de l'image courante. Ainsi, s'il n'y a pas ou peu de mises en correspondance 3D-2D entre les points du modèle et ceux de l'image courante, la méthode de SLAM contraint se comporte alors comme un

SLAM classique. La figure 5.1 illustre un cas où la mise en correspondance de points est difficile notamment à cause des conditions d'éclairage.

(a) (b)

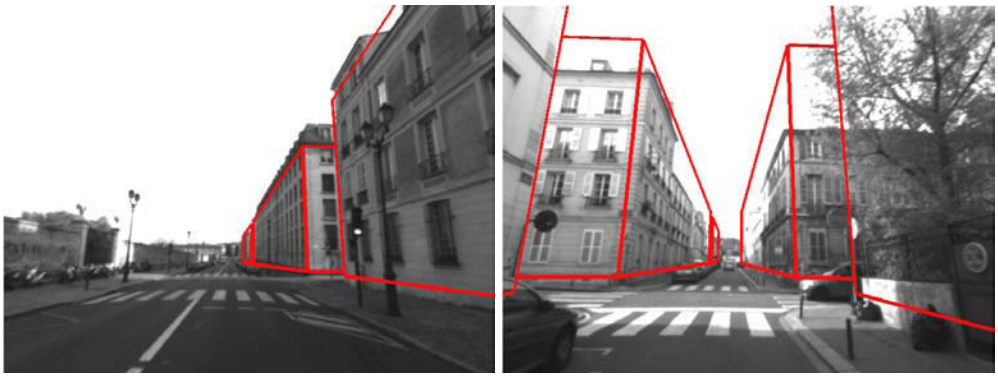
FIGURE 5.1 – Deux images d'une même scène pour lesquelles la mise en correspondance de points d'intérêt est difficile à cause des conditions d'éclairage qui varient entre l'image (a) et l'image (b).

De manière identique, le point faible de l'approche de localisation unifiant SLAM avec un suivi basé modèle géométrique, est l'appariement 3D-2D entre les segments 3D et les contours dans l'image. Nous avons vu que cette approche présente une certaine robustesse aux imperfections de modèles, par exemple un modèle reconstruit avec un capteur Kinect. Cependant, elle ne convient pas à tout type de modèle approximatif, notamment, s'il représente uniquement une boîte englobante de l'objet, la méthode ne fonctionne pas.

Par exemple, sur des modèles 3D de villes issus de Système d'Information Géographique [1] (SIG), cette méthode utilisée pour la localisation en milieu urbain échoue. Dans ce contexte, il est difficile de mettre en correspondance des segments 3D du modèle des bâtiments avec des contours dans les images (voir la figure 5.2). En effet, sur la façade d'un bâtiment il y a plusieurs structures parallèles qui sont susceptibles de générer des contours parallèles dans les images : par exemple dans un bâtiment, les segments correspondants à l'angle de la gouttière et à l'angle du mur peuvent être aisément confondus dans les images. De plus, ce type de modèles est peu détaillé et ne permet donc pas de fournir suffisamment de contraintes liées aux associations 3D-2D entre les segments extraits du modèle et les contours détectés dans les images.

Pour traiter ces cas, nous proposons dans ce chapitre une nouvelle forme de SLAM contraint reposant sur des contraintes géométriques plus souples. Elles considèrent un alignement entre les primitives 3D reconstruites par le SLAM et les surfaces du modèle géométrique. Nous parlerons alors de SLAM contraint avec

1. Modèles de villes de la base IGN basse définition.



(a) (b)

FIGURE 5.2 – Projection du modèle 3D de ville dans des images. Le modèle 3D SIG de l'environnement peut être proche (a) ou éloigné (b) de la géométrie réelle de la scène parcourue.

mise à jour du modèle.

Nous avons vu dans la section 2.3 du chapitre 2 que des méthodes permettent de mettre à jour un modèle 3D géométrique d'un objet d'intérêt. Pour cela ces méthodes utilisent principalement la rétro-projection des observations 2D de points d'intérêts sur le modèle (Platonov *et al.* (2006); Sourimant *et al.* (2007)). L'inconvénient majeur de ces méthodes est que la rétro-projection est réalisée sur une seule vue ; ce qui pose problème en cas d'occultation de l'objet et génère souvent des points peu précis. En effet, en cas d'occultation, tous les points 2D correspondant à des éléments 3D se trouvant entre la caméra et l'objet d'intérêt, seront projetés sur le modèle et seront associés à tord à l'objet. Il s'avère alors préférable de reconstruire ces points 3D par triangulation sur plusieurs vues, puis de les associer au modèle pour les contraindre. Comme proposé dans Lothe *et al.* (2009) la reconstruction multi-vues du SLAM peut être utilisée pour identifier les primitives 3D susceptibles d'appartenir au modèle. Pour cela ils proposent un critère de proximité entre un point du SLAM et une surface du modèle. La trajectoire de la caméra par rapport au modèle est ensuite réestimée en faisant une optimisation non linéaire contrainte uniquement sur les points associés au modèle. Cette optimisation consiste à ajouter des contraintes géométriques sur les caméras observant ces points : les rayons de vues sont contraints à s'intersecter sur le modèle. A noter que ces points 3D ne sont pas explicitement optimisés.

Cette approche connaît cependant plusieurs inconvénients :

- ▷ L'objet doit être visible dans toutes les images clés associées à la reconstruction SLAM.
- ▷ Les points de l'environnement sont exclus de l'optimisation. Dans les images où l'objet est petit, la localisation est peu précise car les points sont mal répartis dans ces images.

▷ La contrainte utilisée (voir la section 5.2.1) modifie la structure creuse de la matrice Hessienne et est donc moins adaptée au temps réel.

Pour répondre à ces problèmes, nous proposons d'intégrer les primitives de l'environnement via notre cadre de SLAM contraint. Le principe repose sur un ajustement de faisceaux, dans lequel certaines primitives reconstruites par le SLAM sont associées au modèle de l'objet d'intérêt et d'autres à l'environnement. Notre positionnement par rapport aux travaux de Lothe *et al.* (2009), repose sur un cadre générique permettant une localisation, d'une part plus robuste et précise, du fait de la prise en compte de l'environnement et d'autre part temps réel par la formulation des contraintes géométriques.

5.2 Ajustement de faisceaux contraint aux plans pour le pSLAM

5.2.1 Contraintes planaires

Dans un premier temps, nous présentons différentes contraintes planaires qui peuvent être utilisées dans notre contexte pour combiner les informations du modèle et la géométrie multi-vues en un même terme. Ensuite, nous nous focalisons sur une contrainte qui s'avère plus adaptée aux performances temps réel dans le cadre d'un ajustement de faisceaux car elle permet de conserver la structure creuse des matrices Hessienne et Jacobienne. A noter que dans cette section, seules les contraintes exprimées en pixels sont considérées.

5.2.1.1 Fonctions de coût existantes

Quelques travaux antérieurs utilisent les bénéfices liés au fait que la structure observée soit planaire par morceaux [2] pour améliorer la qualité de l'algorithme SLAM, par exemple Bartoli et Sturm (2003); Simon et Berger (2002); Szeliski et Torr (1998). Une des possibilités est d'utiliser la relation qui lie deux observations d'un même plan π par une homographie H. Cette contrainte a préalablement été utilisée par Simon et Berger (2002) et Szeliski et Torr (1998) respectivement dans les processus d'estimation de pose et d'appariement de primitives 2D. Cette relation peut également être incorporée dans un processus de raffinement linéaire Xu *et al.* (2000) ou encore dans le processus de raffinement non linéaire de la reconstruction du SLAM à travers la fonction de coût décrite par l'équation suivante :

$$\mathcal{E}_M\left(\left\{(\mathsf{R},\mathbf{t})_{p\to p+1}\right\}_{p=1}^{N_c-1}\right) = \sum_{i\in\mathcal{M}}\sum_{j\in\mathcal{A}_i}\sum_{k\in\mathcal{A}_i}^{k\neq j} d^2(\mathbf{q}_{i,j}, \mathsf{H}_{j,k}^{\pi_i}\mathbf{q}_{i,k}), \tag{5.1}$$

2. Pour plus de détails sur les méthodes d'optimisation plan par morceaux nous renvoyons le lecteur à la thèse d'Adrien Bartoli (Bartoli (2003))

où $\mathsf{H}_{j,k}^{\pi_i}$ est l'homographie induite par l'observation du plan π_i à travers les poses j et k de la caméra. Nous notons N_c le nombre de caméra, $\mathcal{M}$ l'ensemble des indices des points 3D associés aux plans (voir la section 5.4) et $\mathcal{A}_i$ l'ensemble des indices des poses de la caméra observant le i^e point 3D. L'équation (5.1) dépend directement du déplacement de la caméra entre des images $(\mathsf{R}, \mathbf{t})_{j \to k}$. Le fait d'utiliser les contraintes homographiques dans un processus d'ajustement de faisceaux introduit un grand nombre de résidus : pour chaque point 3D $\mathbf{Q}_i$ avec $i \in \mathcal{M}$, précédemment reconstruit, $n_i!$ résidus doivent être ajoutés, chacun dépendant de deux poses de la caméra. De plus, les dérivées analytiques de cette fonction de coût ne sont pas triviales. Notons que cette fonction de coût n'optimise pas explicitement les points 3D. Ils doivent être retriangulés dans un second temps, à partir des poses raffinées de la caméra.

Lothe *et al.* (2009) proposent une alternative pour utiliser les contraintes planaires dans un processus de raffinement non linéaire. Le principe est de faire converger les rayons de vues pour qu'ils se croisent sur les plans du modèle. Pour chaque point 3D $\mathbf{Q}_i$, précédemment reconstruit, ils imposent des contraintes basées modèle en rétro-projetant ces observations n_i sur la surface du modèle à partir des différents points de vue. Ainsi, n_i points $\{\mathbf{Q}_{i,j}\}_{j=1}^{n_i}$ temporaires sont générés et contraints à se déplacer sur la surface du modèle. En pratique pour le raffinement, ils proposent de ne conserver qu'un seul point parmi ceux qui sont générés. Le choix de ce point porte sur le barycentre des points 3D générés par rétro-projection, $\overline{\mathbf{Q}}_i = \frac{1}{n_i} \sum_{j=1}^{n_i} \mathbf{Q}_{i,j}$. Ce dernier est finalement reprojeté dans les images qui lui sont associées pour mesurer les erreurs de reprojection. La fonction de coût décrite par l'équation suivante est alors minimisée :

$$\mathcal{E}_M \left(\{\mathsf{R}_j, \mathbf{t}_j\}_{j=1}^{N_c} \right) = \sum_{i \in \mathcal{M}} \sum_{j \in A_i} d^2(\mathbf{q}_{i,j}, \mathsf{P}_j \overline{\mathbf{Q}}_i), \tag{5.2}$$

où P_j est la matrice de projection associée à la j^e pose de la caméra. Chaque résidu dépend de n_i poses de camera. A noter que les dérivées qui lui sont associées à cette fonction de coût sont calculées de manière numérique.

5.2.1.2 Fonction de coût proposée

Les deux contraintes décrites précédemment présentent des inconvénients pour une utilisation dans le cas d'un ajustement de faisceaux local : soit elle ajoute un grand nombre de résidus (l'équation (5.1)), soit elle nécessite un calcul de dérivées numérique (l'équation (5.2)).

Afin de disposer d'une solution pouvant s'intégrer dans un ajustement de faisceaux local temps réel, nous proposons ici une nouvelle formulation de la contrainte planaire. Contrairement aux précédentes, celle-ci permet de conserver à la fois :

- ▷ un calcul des dérivés analytique très simples ;
- ▷ un nombre de résidus identique à celui de l'ajustement de faisceaux classique ;

▷ une structure par blocs identique à celui de l'ajustement de faisceaux classique (voir la section 5.6).

La fonction de coût résultante est très proche de l'erreur de reprojection utilisée dans l'ajustement de faisceaux pour les points de l'environnement : il y a une cohérence [3] de traitement entre les points de l'environnement et les points du modèle.

L'idée principale de cette fonction de coût est qu'un point 3D $\mathbf{Q}_i$ appartenant à un plan π_i ne possède que deux degrés de liberté (voir le tableau 5.1). Soit M^{π_i} la matrice de transfert entre le repère du plan π_i et le repère monde, on a alors la relation suivante $\mathbf{Q}_i = \mathsf{M}^{\pi_i}\mathbf{Q}_i^{\pi_i}$, où $\mathbf{Q}_i^{\pi_i} = (X^{\pi_i}, Y^{\pi_i}, 0, 1)^\mathsf{T}$ et (X^{π_i}, Y^{π_i}) sont les coordonnées de $\mathbf{Q}_i$ dans le repère du plan π_i. La fonction de coût $\mathcal{E}_M$ résultante, utilisée pour les contraintes planaires dans l'ajustement de faisceaux contraint est décrite par l'équation :

$$\mathcal{E}_M\left(\{\mathsf{R}_j, \mathbf{t}_j\}_{j=1}^{N_c}, \{\mathbf{Q}_i^{\pi_i}\}_{i\in\mathcal{M}}\right) = \sum_{i\in\mathcal{M}}\sum_{j\in\mathcal{A}_i} d^2(\mathbf{q}_{i,j}, \mathsf{P}_j\mathsf{M}^{\pi_i}\mathbf{Q}_i^{\pi_i}). \tag{5.3}$$

Notons que $\mathcal{M}$ représente l'ensemble des indices des points 3D $\mathbf{Q}_i$ associés [4] au modèle, $\mathbf{q}_{i,k}$ correspond à l'observation du point 3D $\mathbf{Q}_i$ dans la caméra C_k et $\mathcal{A}_i$ est l'ensemble des indices de caméras observant ce point.

	Primitives reconstruites	
	Associées à l'environnement ($\mathcal{E}_E$)	Associées aux plans du modèle ($\mathcal{E}_M$)
pSLAM	**3-DoF**	**2-DoF**

TABLE 5.1 – **Unification pSLAM et localisation avec mise à jour du modèle.** Ajustement de faisceaux fondé sur des points 3D contraints aux plans d'un modèle géométrique 3D. Les contraintes sont exprimées en fonction du nombre de degrés de liberté (DoF) attribués aux différentes primitives.

Notons cependant que, dans le cas d'applications ne nécessitant pas de traitement temps réel, la contrainte homographique (l'équation (5.1)) peut aisément être intégrée dans notre cadre d'ajustement de faisceaux contraint aux plans. Celle-ci présente notamment l'avantage de posséder un bassin de convergence plus large permettant de mieux adresser le problème d'ajustement de faisceaux global. Ce type d'applications sortant du contexte premier de cette thèse, nous avons reporté nos contributions sur ce sujet en annexe.

3. Nous présenterons en annexe des résultats comparatifs entre cette fonction de coût et celle qui utilise les contraintes homographiques (l'équation (5.1)).

4. En pratique, les points 3D reconstruits n'appartiennent pas exactement aux plans du modèle (problème lié à la précision de l'estimation des points 3D, plans 3D du modèle pas forcément parfaits). Une étape préliminaire est alors requise pour projeter chaque point 3D $\mathbf{Q}_i$ sur le plan auquel il a été associé $\{\pi_i\}_{i\in\mathcal{M}}$ (voir la section 5.4 pour plus de détails).

5.3 Ajustement de faisceaux contraint aux plans pour le sSLAM

Nous montrons ici que l'approche présentée précédemment peut être généralisée à d'autres primitives que les points. Pour des scènes peu texturées, il y a peu de points d'intérêt détectés ; il est alors préférable de privilégier les segments comme primitives. En effet, pour ce genre de scène les contours sont généralement plus facilement détectables. Pour cela l'objectif est donc de réaliser un SLAM basé segments (sSLAM) contraint aux plans du modèle. Dans cette section nous commencerons, pour le lecteur intéressé, à faire un état de l'art des méthodes de sSLAM et nous proposerons par la suite une adaptation de notre ajustement de faisceaux contraint aux plans pour ces primitives (segments). A noter qu'il s'agit ici d'une étude préliminaire dans laquelle nous nous affranchissons des problèmes de détection et de mise en correspondance inter-images de segments.

5.3.1 Etat de l'art du sSLAM

Nous nous intéressons ici uniquement aux méthodes de sSLAM monoculaire temps réel. Dans ce contexte, la détection de segments est généralement réalisée en deux étapes qui sont la détection de contours (Canny (1986); Gates *et al.* (2005)) et l'extraction de segments (Wang *et al.* (2008); Von Gioi *et al.* (2010)). Gee et Mayol-Cuevas (2006) proposent une solution de sSLAM reposant sur une méthode d'asservissement visuel virtuel (AVV) (voir Comport *et al.* (2006)) pour estimer les poses de la caméra. De plus, un filtre de Kalman sans parfum (UKF pour *Unscented Kalman Filter*) est utilisé pour initialiser de nouveaux segments 3D et pour estimer leurs profondeurs. Smith *et al.* (2006) ont également proposé un sSLAM temps réel qui est une extension du système pSLAM utilisant le filtre de Kalman étendu (EKF pour *Extended Kalman Filter*) pour des appariements de droites.

La difficulté majeure du sSLAM, est la mise en correspondance inter-images de ces segments de droites. Gee et Mayol-Cuevas (2006); Smith *et al.* (2006) n'utilisent pas de descripteur de segments, pour établir ces correspondances. Dans leurs systèmes, le segment 3D projeté correspond, tout simplement, au plus proche segment 2D détecté dans l'image. Par conséquent, de mauvaises correspondances se produisent souvent dans les scènes complexes incluant de nombreux segments. Zhang et Koch (2011) présentent un sSLAM qui utilise l'EKF pour l'estimation de la pose de la caméra et une méthode de reconnaissance de segment basée sur le descripteur de segments MSLD (Wang *et al.* (2009)). Il s'agit d'un descripteur basé sur des histogrammes d'angles, robuste mais coûteux en temps de calcul. Le MSLD est alors utilisé uniquement en cas d'échec du suivi pour la relocalisation de la caméra. Hirose et Saito (2012) proposent également un sSLAM utilisant un descripteur rapide de segments (LEHF).

Les systèmes de sSLAM décrits précédemment utilisent un *filtering based*

SLAM. A notre connaissance, la seule approche de type *keyframe based* utilisant des segments est celle proposée par Klein et Murray (2008). Il ne s'agit pas d'un sSLAM à proprement parler car les segments (*edgelet*) ne viennent qu'en support aux points indispensables pour initialiser le processus. Ils démontrent que grâce à l'intégration des segments, une amélioration de la robustesse au flou de bouger de la caméra est avérée.

5.3.2 Représentation des segments 3D et fonction de coût

Nous abordons ici le problème de l'ajustement de faisceaux, impliquant des poses de caméras et de segments. Dans notre cas la paramétrisation des segments 3D doit mener à une mise à jour d'un nombre minimal de paramètres pour assurer une estimation optimale dans l'ajustement de faisceaux. La paramétrisation des segments 3D dans l'espace est liée à celle des droites. Différentes représentations de droites sont présentées dans Hartley et Zisserman (2000); Bartoli et Sturm (2005) mais ne sont généralement pas adaptées à l'optimisation non-linéaire. De plus, le problème principal des droites est qu'elles ne sont pas localisées et donc difficiles à mettre en correspondance. Par ailleurs, un segment peut être représenté par ses deux extrémités (6 degrés de liberté) mais ces extrémités sont imprécises donc également difficiles à mettre en correspondance. Une alternative est de représenter un segment par son centre et sa direction avec une longueur fixe (Eade et Drummond (2006a)). Il s'agit d'une représentation non minimale à 6 degrés de liberté, souvent utilisée en raison de sa simplicité.

Nous avons choisi la paramétrisation proposée par Klein et Murray (2008) qui est la facilement adaptable au formalisme utilisé dans cette thèse. La paramétrisation choisie consiste également à représenter un segment 3D par son point milieu et sa direction. Ils proposent de paramétriser chaque segment par une matrice de pose $\mathsf{M}^{\mathcal{E}} = \left(\mathsf{R}^{\mathcal{E}} \mid \mathsf{t}^{\mathcal{E}}\right)$, où $\mathcal{E}$ est le repère local associé au segment. Dans ce repère, le centre du segment est situé à l'origine et la direction du segment est parallèle à l'axe des z. Comme présenté sur la figure 5.3, l'axe des x est aligné de manière à pointer dans la direction du gradient local (du coté le plus sombre au coté le plus clair) dans l'image dans laquelle il a été observé pour la première fois. Avec ses 6-DoF cette représentation n'est donc pas minimale, mais permet de ne mettre à jour que 4 paramètres lors de l'optimisation : Klein et Murray (2008) proposent de ne pas optimiser la rotation autour de l'axe z et la translation le long de cet axe. Cela revient alors a une estimation de 4-DoF.

L'erreur de reprojection d'un segment 3D

Pour réaliser l'ajustement de faisceaux nous avons besoin d'un critère à optimiser. Nous choisissons l'erreur de reprojection proposée par Klein et Murray (2008) et définie par deux distances dans l'image entre la reprojection de deux points du segment 3D et la droite 2D qui lui est associée (voir la figure 5.5). Deux points 3D

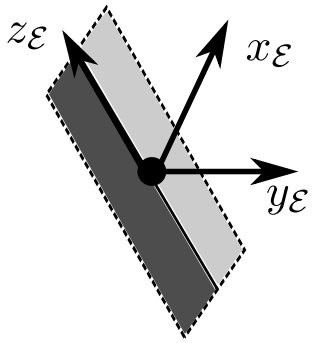


FIGURE 5.3 – Le repère local $\mathcal{E}$ associé à un segment 3D. Ce repère est centré sur le point milieu du segment 3D.

$\mathbf{Q}^1 = \mathsf{M}^\mathcal{E}\ \mathbf{Q}^{1\mathcal{E}}$ et $\mathbf{Q}^2 = \mathsf{M}^\mathcal{E}\ \mathbf{Q}^{2\mathcal{E}}$ sont symétriquement choisis sur le segment de sorte que leurs projections $\mathbf{s}^1$ et $\mathbf{s}^2$ dans l'image soient à 5 pixels du centre (pour plus de détails voir Klein et Murray (2008)). Les coordonnées de ces deux points 3D dans le repère local du segment $\mathcal{E}$ sont exprimées par de la manière suivante :

$$\mathbf{Q}^{1\mathcal{E}} = \begin{pmatrix} 0\ 0\ \dfrac{-5}{|\mathbf{d}|}\ 1 \end{pmatrix}^\mathsf{T},\ \mathbf{Q}^{2\mathcal{E}} = \begin{pmatrix} 0\ 0\ \dfrac{5}{|\mathbf{d}|}\ 1 \end{pmatrix}^\mathsf{T}, \tag{5.4}$$

où $\mathbf{d}$ est la projection dans l'image du vecteur directeur du segment. Les projections de ces deux points sont définies par :

$$\mathbf{s}^1 = \mathsf{P}\ \mathsf{M}^\mathcal{E}\ \mathbf{Q}^{1\mathcal{E}},\ \mathbf{s}^2 = \mathsf{P}\ \mathsf{M}^\mathcal{E}\ \mathbf{Q}^{2\mathcal{E}}, \tag{5.5}$$

où P la matrice de projection de la caméra et $\mathsf{M}^\mathcal{E}$ la matrice de transfert entre le repère local $\mathcal{E}$ associé au segment 3D et le repère monde (voir la figure 5.4).

Pour un segment 3D, les erreurs ε^n avec $n = 1, 2$ sont les deux distances suivant la normale au vecteur directeur (voir la figure 5.5) entre les points $\mathbf{s}^1, \mathbf{s}^2$ et la droite $\mathbf{a} + \lambda\mathbf{b}$ mesurée dans l'image. Ces distances sont décrites par l'équation suivante :

$$\varepsilon^n = \mathbf{n}.\left(\mathbf{a} + \left(\frac{\mathbf{d}.\mathbf{s}^n - \mathbf{d}.\mathbf{a}}{\mathbf{d}.\mathbf{b}}\right)\mathbf{b} - \mathbf{s}^n\right), \tag{5.6}$$

où $\mathbf{d}$ et $\mathbf{n}$ unitaires, correspondent respectivement à la projection dans l'image du vecteur directeur du segment 3D et à sa normale. Notons que cette erreur de reprojection est exprimée en pixels. Par la suite, cette distance point à droite sera notée $d_{dp}\left(\left(\mathbf{a} + \lambda\mathbf{b}\right), \mathbf{s}\right)$.

Fonction de coût pour les segments de l'environnement

Pour une scène composée de N_s segments 3D et de N_c caméras, la fonction de coût de l'ajustement de faisceaux des segments 3D est donnée par l'équation :

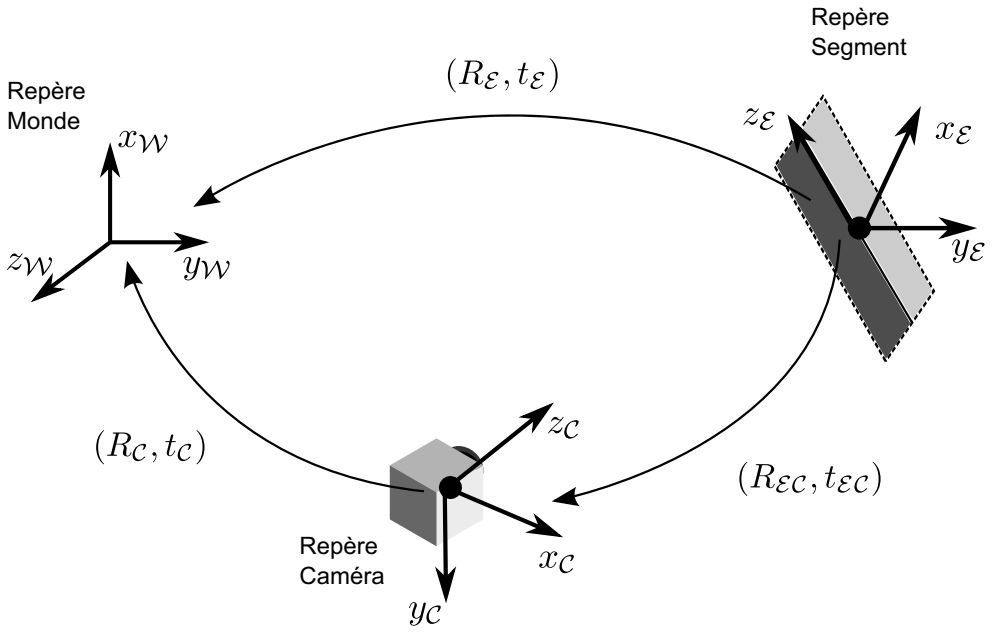


FIGURE 5.4 – Paramétrisation des segments de l'environnement.

$$\mathcal{E}_E\left(\left\{\mathsf{R}_j, \mathbf{t}_j\right\}_{j=1}^{N_c}, \left\{\boldsymbol{\mu}_p^{\mathcal{E}_i}\right\}_{i=1}^{N_s}\right) = \sum_{i=1}^{N_s}\sum_{j\in\mathcal{S}_i}\sum_{n=1}^{2} d_{dp}^2\left((\mathbf{a}+\lambda\mathbf{b})_{i,j}, \mathsf{P}_j\mathsf{M}^{\mathcal{E}_i}\,\mathbf{Q}^{n\mathcal{E}_i}\right), \quad (5.7)$$

avec $\mathbf{Q}^{n\mathcal{E}_i}$ correspondant au point $\mathbf{Q}^{n\mathcal{E}}$ associé au segment i (voir paragraphe précédent) et $\mathcal{S}_i$ l'ensemble des indices de caméras observant ce segment. $\boldsymbol{\mu}^\mathcal{E}$ est le vecteur d'état défini par $\mathsf{M}^\mathcal{E} = \exp(\boldsymbol{\mu}^\mathcal{E})$, et dont les trois premiers et les trois derniers éléments correspondent respectivement à la translation et la rotation par rapport aux axes x, y et z du repère $\mathcal{E}$. Sur ces six éléments, seules les éléments $p = 1, 2, 4, 5$ sont considérés dans l'optimisation de la pose $\mathsf{M}^{\mathcal{E}_i}$ de chaque segment 3D : ceux qui n'ont pas d'influence sur l'erreur de reprojection (translation le long de l'axe z et rotation autour de cet axe) sont ignorés.

5.3.3 Contraintes Planaires

Dans le cas des scènes peu texturées, nous venons de voir qu'il est possible de reconstruire par une méthode de type sSLAM des segments 3D. Ces derniers peuvent alors être contraints par les plans du modèle, comme pour le pSLAM. Le sSLAM contraint est présenté ici comme une extension du pSLAM contraint pour lequel on décrit le formalisme théorique que l'on propose. Rappelons que cette étude préliminaire, vérifiant les fondements de l'approche d'ajustement de faisceaux contraint aux plans du modèle, dans le cas de primitives segments, ne portera que sur des données simulées.

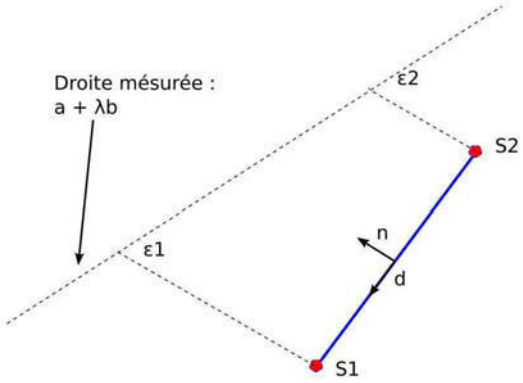


FIGURE 5.5 – **Erreur de reprojection d'un segment 3D.** Elle est obtenue par les distances ε^n suivant la normale au vecteur directeur entre les points $\mathbf{s}^1, \mathbf{s}^2$ et la droite d'équation $\mathbf{a} + \lambda\mathbf{b}$ associée au contour détecté dans l'image.

Fonction de coût proposée

Nous présentons ici une contrainte planaire qui peut être utilisée dans notre contexte pour combiner les informations du modèle et la géométrie multi-vues en un même terme pour des segments. Nous verrons par la suite que cette contrainte est adaptée aux performances temps réel dans le cadre d'un ajustement de faisceaux car elle permet de conserver la structure creuse des matrices Hessienne et Jacobienne. Nous proposons ici une formulation pour les segments proche de celle du pSLAM contraint au plans, permettant de réduire le nombre de degrés de liberté d'un segment i (associé au repère $\mathcal{E}_i$) appartenant à un plan π_i. L'idée principale pour cette fonction de coût est de construire un nouveau repère $\mathcal{E}_i^{'}$ en appliquant à $\mathcal{E}_i$ une rotation $\mathsf{R}_z^{\mathcal{E}_i}$ autour de son axe z. On peut alors écrire :

$$\begin{aligned} \mathsf{M}^{\mathcal{E}_i^{'}}\,\mathbf{Q}^{n\mathcal{E}_i} &= \mathsf{M}^{\mathcal{E}_i}\,\mathsf{R}_z^{\mathcal{E}_i}\,\mathbf{Q}^{n\mathcal{E}_i} \\ &= \mathsf{M}^{\mathcal{E}_i}\,\mathbf{Q}^{n\mathcal{E}_i}, \end{aligned} \tag{5.8}$$

avec $\mathbf{Q}^{n\mathcal{E}_i}$ défini comme dans la section précédente. A noter que la rotation $\mathsf{R}_z^{\mathcal{E}_i}$ n'a pas d'impact sur les points $\mathbf{Q}^{n\mathcal{E}_i}$ car ils sont situés sur l'axe z du repére $\mathcal{E}$. Nous choisissons $\mathsf{R}_z^{\mathcal{E}_i}$ telle que l'axe y du nouveau repère $\mathcal{E}_i^{'}$ soit normal au plan π_i. Dans ce nouveau repère, ainsi construit, le segment i n'a plus que deux degrés de liberté (voir le tableau 5.2) qui sont la translation le long de l'axe x et la rotation autour de l'axe y. L'optimisation du segment i sur le plan π_i sera alors effectuée dans ce repère $\mathcal{E}_i^{'}$. Pour une scène composée de N_s segments 3D, de N_c caméras, la fonction de coût de l'ajustement de faisceaux de segments contraint aux plans est donc donnée par l'équation :

$$\mathcal{E}_M\left(\{\mathsf{R}_j,\mathbf{t}_j\}_{j=1}^{N_c},\left\{\boldsymbol{\mu}_r^{\mathcal{E}_i'}\right\}_{i=1}^{N_s}\right)=\sum_{i=1}^{N_s}\sum_{j\in\mathcal{S}_i}\sum_{n=1}^{2}d_{dp}^2\left(\left(\mathbf{a}+\lambda\mathbf{b}\right)_{i,j},\mathsf{P}_j\mathsf{M}^{\mathcal{E}_i'}\,\mathbf{Q}^{n\mathcal{E}_i}\right). \tag{5.9}$$

Seulement deux paramètres $r=1$ et $r=5$ sont considérés dans l'optimisation de la pose $\mathsf{M}^{\mathcal{E}_i'}$ de chaque segment 3D contraint au plan : ceux qui influent sur l'erreur de reprojection (translation le long de l'axe x et rotation autour de l'axe y). En pratique, les segments 3D reconstruits n'appartiennent pas exactement aux plans du modèle. Une étape préliminaire est alors requise pour projeter chaque segment 3D i sur le plan π_i auquel il a été associé. Les détails de cette étape sont présentés dans la section 5.4.

	Primitives reconstruites	
	Associées à l'environnement ($\mathcal{E}_E$)	Associées aux plans du modèle ($\mathcal{E}_M$)
sSLAM	**4-DoF**	**2-DoF**

TABLE 5.2 – **Unification sSLAM et localisation avec mise à jour du modèle.** Ajustement de faisceaux fondé sur des segments 3D contraints aux plans d'un modèle géométrique 3D. Les contraintes sont exprimées en fonction du nombre de degrés de liberté (DoF) attribués aux différentes primitives.

5.4 Associations 3D-3D entre les primitives 3D et les plans du modèle

Dans les sections précédentes nous avons présenté différentes fonctions de coût, pour raffiner sur le modèle, une reconstruction initiale de type pSLAM (en minimisant une des équations (5.1) à (5.3)) et une reconstruction de type sSLAM (en minimisant l'équation (5.9)). Pour cela les primitives 3D (points ou segments) reconstruites par le SLAM doivent être préalablement associées aux plans du modèle ou à l'environnent. Dans cette section, nous décrivons comment cette étape préliminaire de classification est réalisée pour décider quels points 3D $\mathbf{Q}_i$ ou segments 3D $\mathbf{S}_i$ de la reconstruction appartiennent au modèle.

L'association entre les points 3D reconstruits et les plans du modèle est effectuée par lancers de rayons à partir des différentes observations $\{\mathbf{q}_{i,j}\}_{j\in A_i}$ de $\mathbf{Q}_i$. Card(A_i) votes sont alors obtenus pour les différents plans et le choix majoritaire est conservé. Une fois la classification effectuée pour chaque point 3D associé à un plan π_i du modèle une projection de ce point sur le plan est réalisée. Le barycentre des intersections entre les lancers de rayons et le plan π_i est sélectionné comme étant la position initiale du point 3D lors de la minimisation de l'équation (5.3).

De manière identique aux points, l'association entre les segments 3D reconstruits et les plans du modèle est effectuée par lancers de rayons. Ils sont réalisés à partir des différentes observations $\{\mathbf{m}_{i,j}\}_{j\in A_i}$ du point milieu du segment $\mathbf{S}_i$.

5.5 Optimisation itérative

Les équations (5.10) et (5.11) donnent respectivement les fonctions de coût de l'ajustement de faisceaux contraint aux plans du modèle, dans le cas du pSLAM et du sSLAM. Nous notons $\mathcal{M}$ l'ensemble des indices des primitives 3D associées au modèle et $\mathcal{U}$ l'ensemble des indices des autres primitives 3D qui constituent l'environnement, avec $\text{card}(\mathcal{M}) + \text{card}(\mathcal{U}) = N$, où N est le nombre total de primitives 3D.

$$\begin{aligned}\mathcal{E}\left(\{\mathsf{R}_j,\mathbf{t}_j\}_{j=1}^{N_c},\{\mathbf{Q}_i\}_{i\in\mathcal{U}},\{\mathbf{Q}_i^{\pi_i}\}_{i\in\mathcal{M}}\right) \;=\; & \\ & \underbrace{\sum_{i\in\mathcal{U}}\sum_{j\in\mathcal{A}_i}\rho\left(d^2(\mathbf{q}_{i,j},\mathsf{P}_j\mathbf{Q}_i),c_1\right)}_{\text{Partie inconnue de l'environnement }(\mathcal{E}_E)} \\ & + \underbrace{\sum_{i\in\mathcal{M}}\sum_{j\in\mathcal{A}_i}\rho\left(d^2(\mathbf{q}_{i,j},\mathsf{P}_j\mathsf{M}^{\pi_i}\mathbf{Q}_i^{\pi_i}),c_2\right)}_{\text{Partie connue de l'environnement }(\mathcal{E}_M)}\end{aligned} \tag{5.10}$$

$$\begin{aligned}\mathcal{E}\left(\{\mathsf{R}_j,\mathbf{t}_j\}_{j=1}^{N_c},\left\{\boldsymbol{\mu}_p^{\mathcal{E}_i}\right\}_{i\in\mathcal{U}},\left\{\boldsymbol{\mu}_r^{\mathcal{E}_i'}\right\}_{i\in\mathcal{M}}\right) \;=\; & \\ & \underbrace{\sum_{i\in\mathcal{U}}\sum_{j\in\mathcal{S}_i}\sum_{n=1}^{2}\rho\left(d_{dp}^2\left((\mathbf{a}+\lambda\mathbf{b})_{i,j},\mathsf{P}_j\mathsf{M}^{\mathcal{E}_i}\,\mathbf{Q}^{n\mathcal{E}_i}\right),c_1\right)}_{\text{Partie inconnue de l'environnement }(\mathcal{E}_E)} \\ & + \underbrace{\sum_{i\in\mathcal{M}}\sum_{j\in\mathcal{S}_i}\sum_{n=1}^{2}\rho\left(d_{dp}^2\left((\mathbf{a}+\lambda\mathbf{b})_{i,j},\mathsf{P}_j\mathsf{M}^{\mathcal{E}_i'}\,\mathbf{Q}^{n\mathcal{E}_i}\right),c_2\right)}_{\text{Partie connue de l'environnement }(\mathcal{E}_M)}\end{aligned} \tag{5.11}$$

Au cours du processus d'optimisation de ces deux équations, les associations entre les primitives 3D et les plans du modèle ainsi que les seuils de rejet de l'estimateur robuste doivent être réestimés pour garantir une convergence optimale.

Les étapes du tableau 5.3 sont alors itérées jusqu'à convergence. Cela permet d'éviter les problèmes liés aux occultations [5] que rencontrent les méthodes de mise à jour du modèle. L'approche itérative permet de remettre en cause les associations 3D-3D. Une étape de triangulation des primitives avec les poses des caméras optimisées est réalisée après chaque itération pour remettre en cause les associations. A noter que dans le cas d'un ajustement de faisceaux local une seul itération suffit car on est généralement proche de la solution. Les associations sont également remises en cause à chaque image clé.

5. Les occultations peuvent être gérées tant qu'elles ne sont pas majoritaires, grâce au rejet des mauvaises associations par l'estimateur robuste.

1. Association des primitives 3D au modèle ou à l'environnement.
2. Projection des primitives 3D sur les plans π_i auxquels elles sont associées.
3. Calcul des seuils de rejets c_1 et c_2 de l'équation (5.10) ou (5.11) .
4. Minimisation de l'équation (5.10) dans le cas du pSLAM ou de l'équation (5.11) dans le cas du sSLAM par l'algorithme de Levenberg-Marquardt (quelques itérations).
5. Triangulation [a] des points 3D $(\mathbf{Q}_i)_{i\in\mathcal{M}}$ pour l'équation (5.10) ou des segments 3D $(\mathbf{Q}_i)_{i\in\mathcal{M}}$ pour l'équation (5.11) avec les poses des caméras estimées.

a. Les primitives 3D de la partie connue de l'environnement doivent être triangulées pour la réestimation des associations des primitives 3D aux plans du modèle.

TABLE 5.3 – Etapes de l'ajustement de faisceaux contraint aux plans du modèle dans le cas de primitives points (pSLAM) et segments (sSLAM).

A noter que cette étape d'association 3D-3D peut être améliorée, nous pouvons notamment citer Larnaout *et al.* (2012). Dans leurs travaux, ils utilisent un SLAM contraint aux plans dans un contexte de localisation de véhicule en milieu urbain. Les modèles utilisés sont les modèles 3D SIG. L'étape d'association est alors modifiée pour prendre en compte de manière statistique les orientations des plans du modèle.

5.6 Structure creuse de l'ajustement de faisceaux contraint aux plans

Cette section, montre que la structure creuse des matrices Hessienne et Jacobienne, associées à l'ajustement de faisceaux, sont conservées malgré l'ajout de la contrainte aux plans, pour les équations (5.10) et (5.11).

Les figures 5.6 et 5.7 (à gauche) représentent respectivement des exemples d'ajustement de faisceaux avec des contraintes planaires, dans le cas du pSLAM et sSLAM. La matrice Jacobienne associée à l'équation (5.10) est très similaire à celle d'un ajustement de faisceaux classique. La seule différence est que les primitives (points ou segments) n'ont pas le même nombre de degrés de liberté suivant qu'elles soient associées à l'environnement ou aux plans du modèle. Ceci implique que la matrice Hessienne a la même structure par blocs que pour l'ajustement de faisceaux dans sa version originale. En effet, la structure de la matrice U est inchangée, V est toujours une matrice diagonale par blocs :

- ▷ cas du pSLAM : V est composée de blocs 3×3 (en rouge foncé sur la figure 5.6) ainsi que de blocs 2×2 (en bleu foncé sur la figure 5.6). Cela implique

que la matrice W est composée de blocs 6×3 (blocs rose sur la figure 5.6) et de blocs 6×2 (bleu clair sur la figure 5.6) ;

▷ cas du sSLAM : V est composée de blocs 4×4 (en rouge foncé sur la figure 5.7) ainsi que de blocs 2×2 (en bleu foncé sur la figure 5.7). Cela implique que la matrice W est composée de blocs 6×4 (blocs rose sur la figure 5.7) et de blocs 6×2 (bleu clair sur la figure 5.7).

Ainsi, comme pour l'ajustement de faisceaux contraint présenté dans les sections 4.2 et 4.3 du chapitre 4, il est possible d'implémenter de manière efficace l'ajustement de faisceaux contraint aux plans, en prenant en compte cette structure par blocs, comme décrit dans Triggs *et al.* (2000).

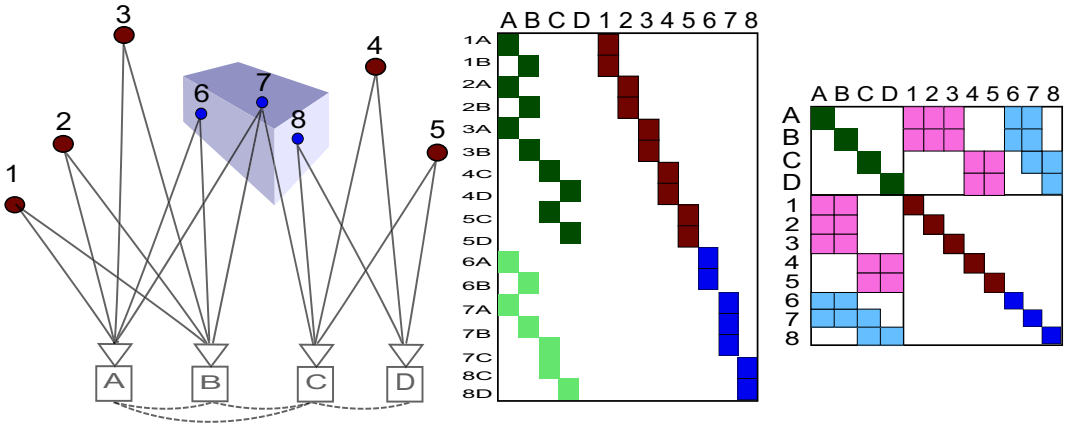


FIGURE 5.6 – A gauche : un exemple d'ajustement de faisceaux contraint aux plans du modèle, pour des primitives de type point. En rouge foncé, les points 3D de la partie inconnue de l'environnement (de 1 à 5) ; en bleu foncé, les points 3D associés à la partie connue de l'environnement (de 6 à 8). Les carrés (de A à D) représentent la trajectoire de la camera. Les lignes noires indiquent l'observabilité d'un point 3D à partir d'une certaine pose de la caméra en mouvement. Au milieu et à droite : les matrices Jacobienne et Hessienne associées.

5.7 Résultats expérimentaux

5.7.1 pSLAM contraint aux plans

5.7.1.1 Évaluation sur des données de synthèse

Dans cette section nous comparons deux algorithmes d'ajustement de faisceaux contraint pour le pSLAM : pBA_M, pBA_M&E [6]. Ils minimisent respectivement les équations (5.3) et (5.10) en suivant la procédure décrite dans le tableau 5.3. Notons

6. p signifie qu'il s'agit de primitives de type points (pSLAM), M signifie que seules les contraintes au modèle (c'est-à-dire de la partie connue de l'environnement) sont utilisées tandis

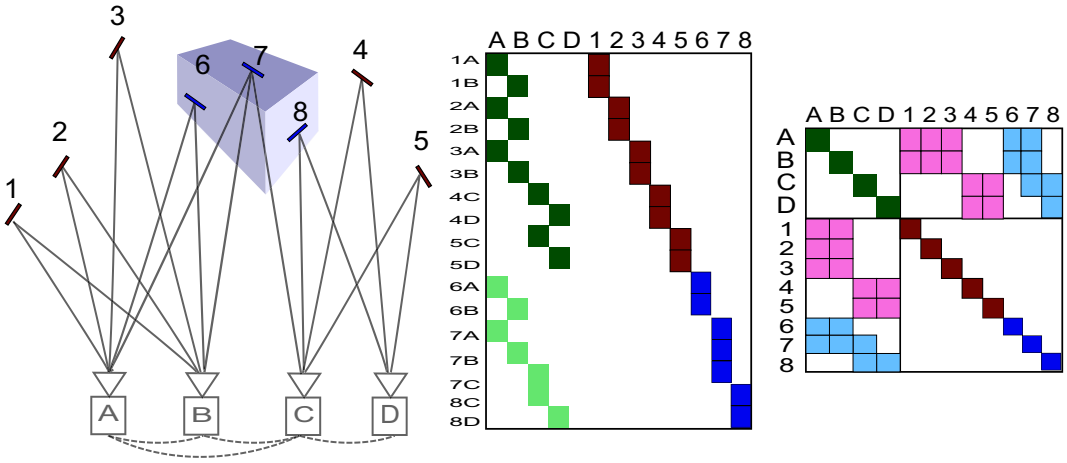


FIGURE 5.7 – A gauche : un exemple d'ajustement de faisceaux contraint aux plans du modèle, pour des primitives de type segment. En rouge foncé, les segments 3D de la partie inconnue de l'environnement (de 1 à 5) ; en bleu foncé, les segments 3D associés à la partie connue de l'environnement (de 6 à 8). Les carrés (de A à D) représentent la trajectoire de la camera. Les lignes noires indiquent l'observabilité d'un segment 3D à partir d'une certaine pose de la caméra en mouvement. Au milieu et à droite : les matrices Jacobienne et Hessienne associées.

que pour pBA_M le M-estimateur de Geman-Mclure est aussi utilisé pour gérer les mauvaises associations entre les points 3D reconstruits et les plans du modèle.

Dans un premier temps une comparaison des trois choix de combinaisons basés sur les seuils de rejets c_1 et c_2 du M-estimateur de la section 3.2.5 est effectuée avec l'algorithme pBA_M&E. Dans un second temps, une comparaison des deux algorithmes est réalisée dans le cas d'un ajustement de faisceaux global.

La séquence « cube »

La comparaison est réalisée sur une séquence de synthèse. La figure 5.8 représente la séquence « cube » qui est composée d'un cube principal partiellement occulté par d'autres petits cubes et situé sur un sol texturé. L'objet d'intérêt, c'est-à-dire pour lequel un modèle 3D est disponible, est le cube principal. L'environnement inconnu est alors composé du sol et des petits cubes situés autour. La trajectoire de la caméra est un cercle de 3 mètres de rayon autour du cube principal d'un mètre de hauteur.

---

que M&E signifie qu'à la fois les contraintes au modèle mais aussi les informations du reste de l'environnement sont prises en compte.

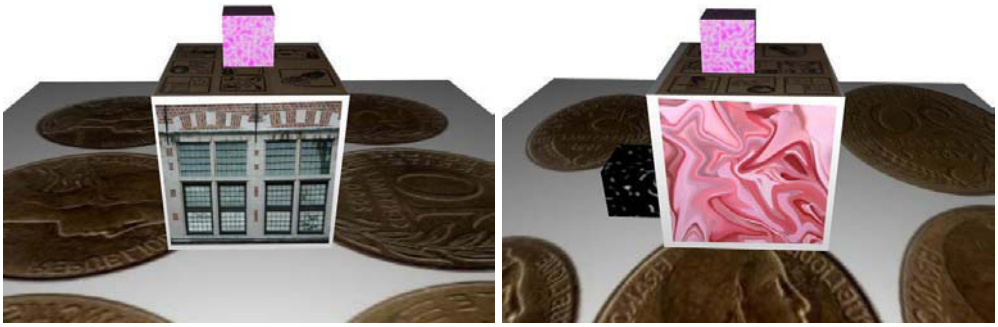


FIGURE 5.8 – Illustration de la séquence « cube ».

Choix de la combinaison

Pour l'algorithme de pBA_M&E nous comparons les trois propositions de combinaisons décrites en section 3.2.5. La reconstruction initiale est obtenue avec l'algorithme de SLAM décrit dans Mouragnon *et al.* (2006). Le repère du monde et l'échelle de la reconstruction sont fixés avec la vérité terrain. Cette reconstruction initiale est alors raffinée en minimisant l'équation (5.10) avec l'une des trois combinaisons, en suivant la procédure décrite dans le tableau 5.3.

La figure 5.9 (à gauche) illustre la distribution des erreurs pour les deux types de résidus : ceux associés aux points de l'environnement et ceux associés aux points du modèle. Les erreurs résiduelles associées aux points du modèle sont plus élevées que celles des points de l'environnement, comme pressenti. Cela explique que la combinaison 2 présente le plus mauvais résultat comme le montre la figure 5.9 (à droite). Comme son seuil de rejet est sous-estimé, la plupart des résidus de la partie modélisée sont rejetés par l'estimateur robuste. L'algorithme pBA_M&E avec la combinaison 1 donne des résultats similaires à pBA_M. Cela montre que cette combinaison tend à ne pas prendre en compte les points de l'environnement. Finalement pBA_M&E avec la combinaison 3 donne les meilleurs résultats car cette combinaison permet de conserver la plupart des primitives associées au modèle. Cela implique qu'à la fois les contraintes au modèle et les contraintes multi-vues des points de l'environnement restent vérifiées. Par la suite seule la combinaison 3 sera utilisée pour l'algorithme pBA_M&E.

Comparaison des deux algorithmes d'ajustement de faisceaux. Nous comparons ici les deux algorithmes pBA_M et pBA_M&E en simulant différentes sources d'erreurs comme celle du recalage initial, de la dérive, *etc*. A partir des poses de caméras de la vérité terrain, un nuage de points 3D éparse est généré par triangulation de points d'intérêt appariés au cours de la séquence. Deux types de perturbations sont alors générées sur cette reconstruction initiale.

▷ Le premier test (TEST RIGIDE) simule des imprécisions du recalage initial

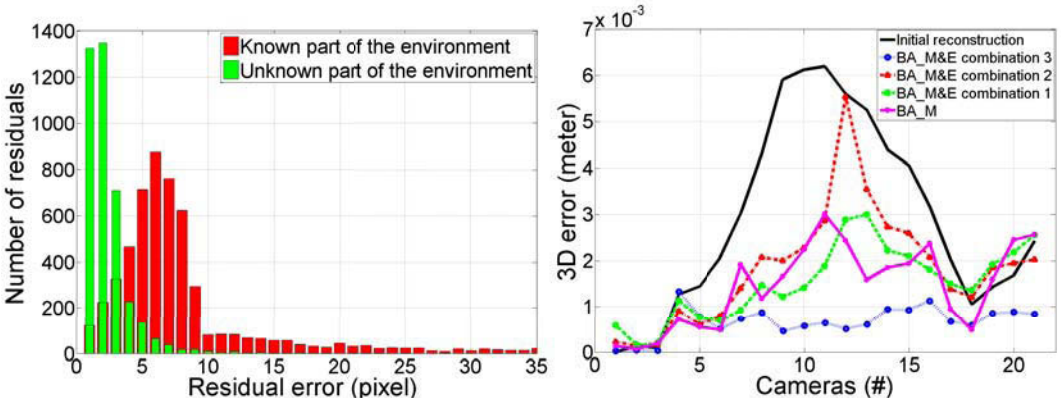


FIGURE 5.9 – Evaluation de la stratégie de pondération proposée. A gauche : distributions des erreurs résiduelles. En vert, (resp. en rouge) la distribution des erreurs résiduelles associées à la composante inconnue (resp. connue). A droite : Erreurs sur les positions de la caméra exprimées en mètres pour les différentes combinaisons.

entre les repères du monde et du modèle. Pour cela, une perturbation rigide est appliquée sur la reconstruction globale (caméras et points 3D).

- ▷ Le deuxième test (TEST NON RIGIDE) simule les erreurs d'estimation du déplacement relatif entre deux caméras. Ces erreurs proviennent de l'estimation, par le SLAM, du déplacement de la caméra en présence de bruit, de valeurs aberrantes, et d'erreurs de calcul numérique, *etc*. Comme ces erreurs ne sont pas constantes dans le temps, une perturbation non rigide de la reconstruction globale est réalisée en perturbant de manière aléatoire les déplacements inter-caméras et en régénérant par la suite un nuage de points 3D par triangulation.

Les 50 tirages aléatoires sont réalisés pour chaque amplitude de perturbation. Cette dernière varie de 1% à 10% du rayon du cercle décrivant la trajectoire de la caméra. Nous appliquons alors les deux algorithmes sur les reconstructions ainsi obtenues. La qualité de la reconstruction finale est mesurée par le RMS 3D sur les positions de la caméra entre celles de la vérité terrain et celles estimées. Les deux algorithmes d'ajustement de faisceaux sont comparés en termes de précision, de vitesse et de fréquence de convergence. La précision est donnée par la valeur du RMS 3D uniquement lorsqu'il y a convergence de l'algorithme. La fréquence de convergence est le pourcentage de tirages pour lesquels le RMS 3D a diminué. La vitesse de convergence est mesurée par l'évolution de l'erreur 3D au cours des itérations successives du Levenberg-Marquardt, pour une perturbation donnée (4% dans nos expériences). Une moyenne a été effectuée sur les 50 tirages aléatoires.

Fréquence de convergence. La figure 5.10 présente les résultats concernant la fréquence de convergence. Les deux algorithmes ont un comportement similaire

pour le TEST RIGIDE et le TEST NON RIGIDE. pBA_M&E a un bassin de convergence plus large que pBA_M. Pour un déplacement d'amplitude 10% lors du TEST NON RIGIDE, pBA_M&E converge approximativement dans tous les cas, alors que pBA_M ne converge jamais.

Précision. L'algorithme pBA_M&E est plus précis que pBA_M pour les deux types de perturbations (rigides et non rigides). Par exemple, pour une amplitude de perturbation de 8% pour le TEST RIGIDE, le RMS 3D de pBA_M&E est de l'ordre de 1% alors que celui de pBA_M est de 3%. Les résultats sont illustrés sur la figure 5.11.

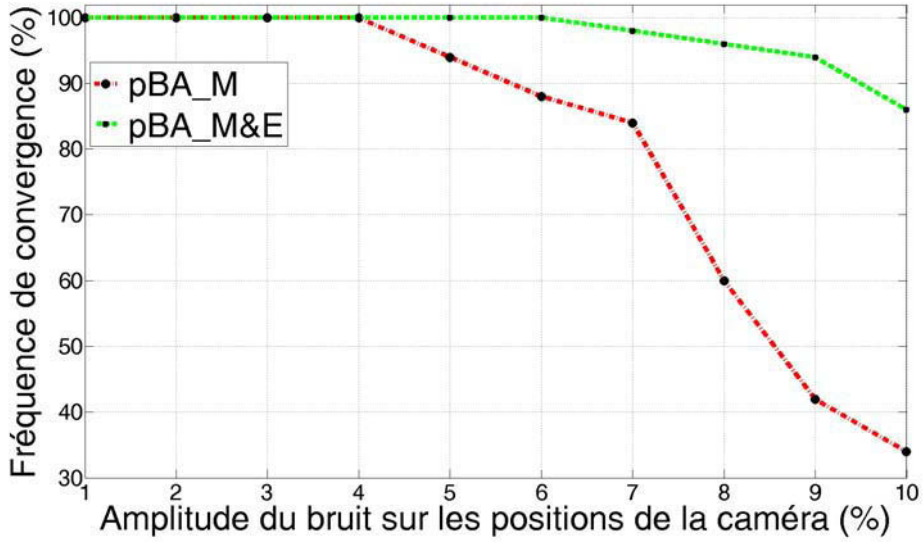
Vitesse de convergence. La figure 5.12 montre que pBA_M&E converge plus vite que l'algorithme pBA_M. Par exemple, pour le TEST RIGIDE, après trois itérations pBA_M&E réduit l'erreur 3D par un facteur de 2.7, et seulement par un facteur de 1.5 pour pBA_M.

Résumé. pBA_M&E est plus performant que pBA_M en termes de précision, de vitesse et de fréquence de convergence. Cela montre que la prise en compte de l'environnement dans l'ajustement de faisceaux contraint aux plans du modèle améliore de manière remarquable les résultats.

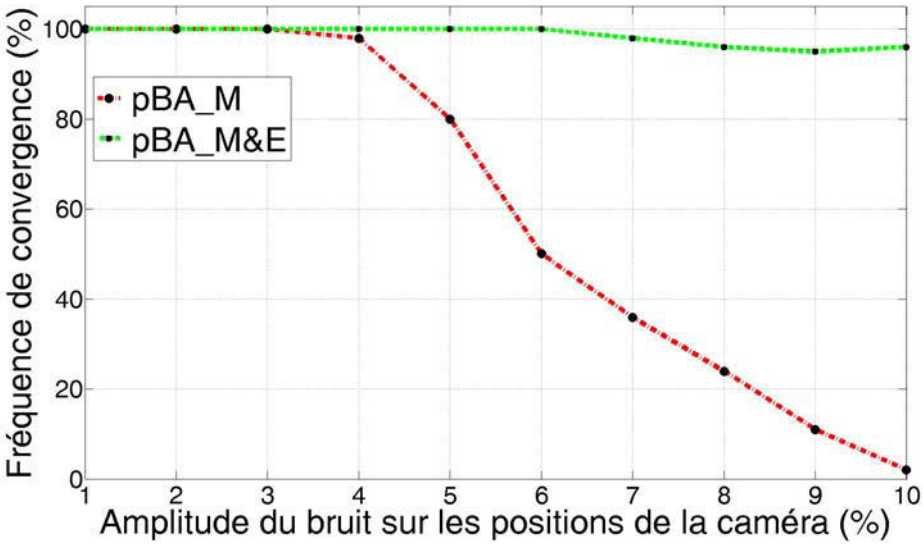
5.7.1.2 Evaluation sur des données réelles

Nous avons évalué les deux algorithmes précédents sur des données réelles pour deux applications : la localisation dans un grand environnement et le suivi d'objet 3D. Deux séquences réelles ont été réalisées avec une caméra IEEE1394 GUPPY fournissant des images (640×480) à 30 images par seconde. Dans le cas de la localisation en grand environnement un ajustement de faisceaux global contraint est mis en œuvre alors que dans le cas du suivi d'objet 3D un ajustement de faisceaux local contraint est évalué. Par mesure de clarté, les évaluations de la localisation en grand environnement sont présentées en annexe. Nous présentons ici les résultats sur la séquence de suivi d'objet. Les conclusions sont les mêmes dans le premier cas.

Nous présentons ici les performances de l'ajustement de faisceaux contraint appliqué au suivi d'objet. Nous utilisons ici un ajustement de faisceaux local dans le pSLAM. L'objet d'intérêt est un modèle réduit de la Citroën C4 de Sebastien Loeb du championnat WRC. Le modèle 3D utilisé dans nos expériences est composé de 1600 triangles (voir la figure 5.15, première ligne). Il n'inclut pas certaines pièces de la voiture comme les roues, l'aileron, les fenêtres, *etc*. Nous avons placé la voiture sur un bureau composé d'un écran et clavier d'ordinateur, de livres, *etc*. Ils constituent la partie inconnue de l'environnement.

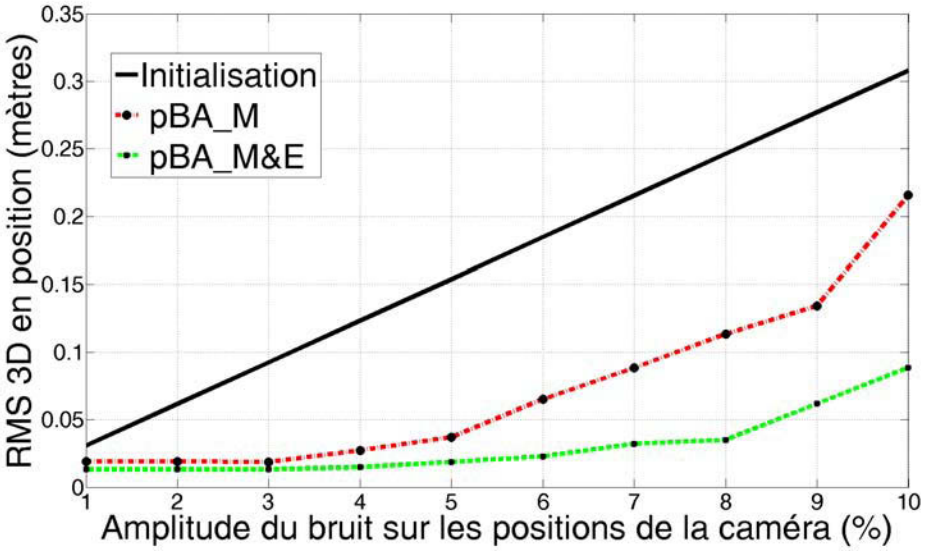


(a)

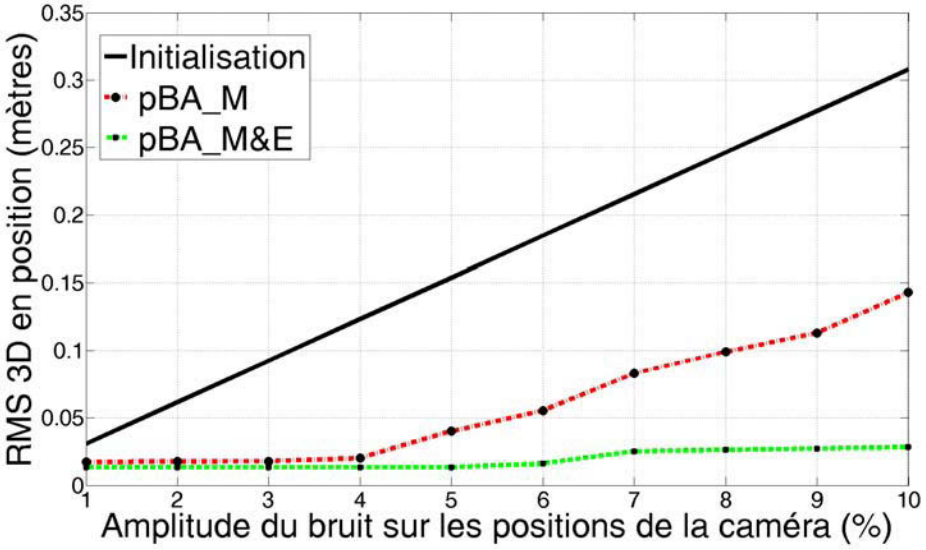


(b)

FIGURE 5.10 – **Evaluation de la fréquence de convergence des algorithmes pBA_M et pBA_M&E.** Résultats obtenus avec les deux algorithmes pour le TEST RIGIDE (a) et le TEST NON RIGIDE (b).

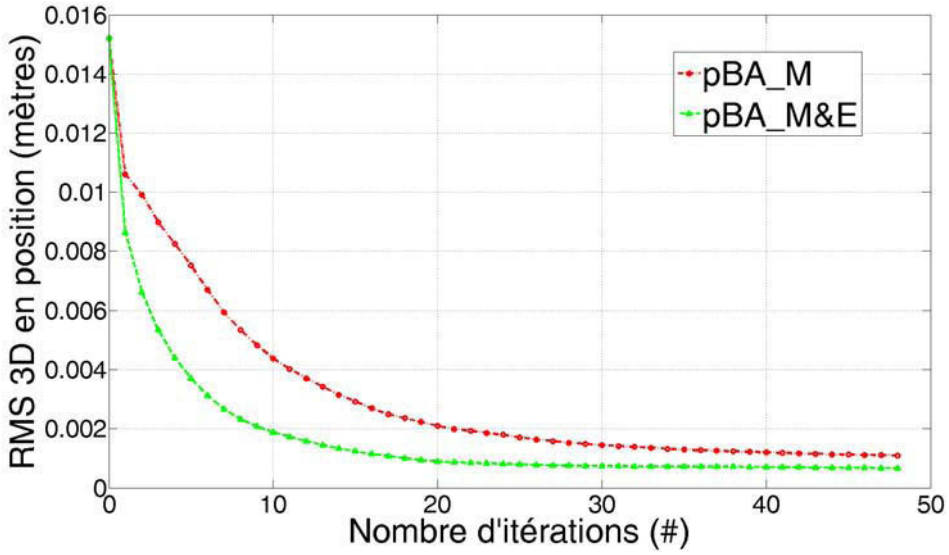


(a)

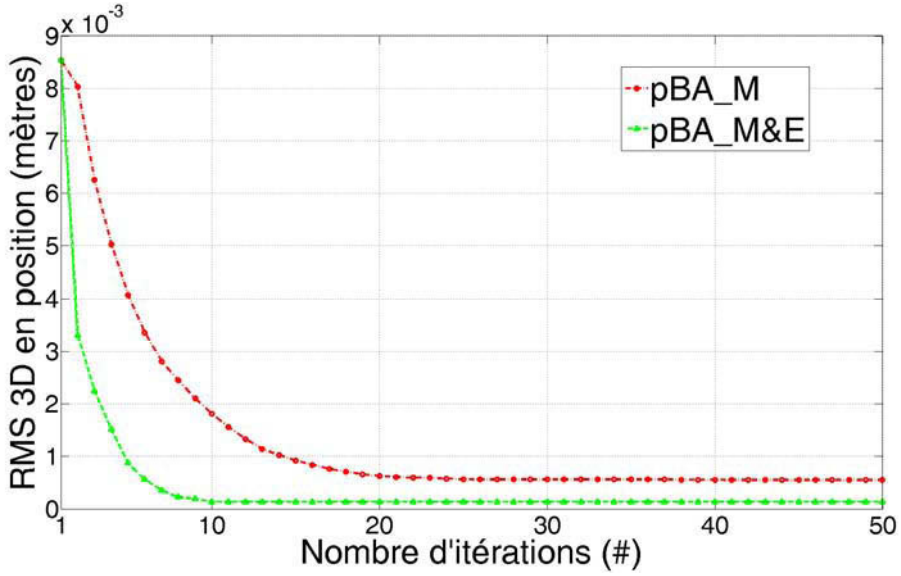


(b)

FIGURE 5.11 – **Evaluation de la précision des algorithmes pBA_M et pBA_M&E.** Résultats obtenus avec les deux algorithmes pour le TEST RIGIDE (a) et le TEST NON RIGIDE (b).



(a)



(b)

FIGURE 5.12 – **Evaluation de la vitesse de convergence des algorithmes pBA_M et pBA_M&E.** Résultats obtenus avec les deux algorithmes pour le TEST RIGIDE (a) et le TEST NON RIGIDE (b).

Algorithmes comparés. Nous comparons dans un contexte de suivi d'objet 3D, les trois algorithmes d'ajustement de faisceaux : pBA_E, pBA_M et pBA_M&E. La comparaison est réalisée sur une séquence difficile qui présente de grandes variations en échelle, des variations de luminosité, des occultations partielles et totales de la voiture, *etc*. Le recalage initial est réalisé en appariant la première image de la séquence avec une image clé qui a été recalée au préalable sur le modèle.

Résultats. La figure 5.13 présente les résultats obtenus par les trois types d'ajustements de faisceaux. Le recalage initial semble précis sur la première image (le modèle se projette précisément) mais en tournant autour de la voiture nous observons que ce n'est pas réellement le cas. L'algorithme d'ajustement de faisceaux pBA_E ne parvient pas à corriger cette imprécision comme illustré sur la figure 5.13 (en haut à gauche) et conserve donc cette erreur au cours de toute la séquence. Les algorithmes d'ajustement de faisceaux contraint pBA_M&E et pBA_M réussissent à corriger cette erreur de recalage après quelques images : l'avant et l'arrière de la voiture se projettent parfaitement dans les images. Notons que l'algorithme pBA_M&E donne de meilleurs résultats que pBA_M quand l'objet d'intérêt est occulté et lorsqu'il est perçu de très loin comme le montre la figure 5.13 (à droite). Il gère avec succès et précision, la localisation de la caméra au cours de toute la séquence. La combinaison des informations fournies par les points du modèle et les points de l'environnement permet donc une localisation plus précise et robuste. La figure 5.14 montre la classification des points réalisée par l'algorithme pBA_M&E au cours de la séquence.

Robustesse à des imprécisions de modèle. Nous montrons ici que la méthode de SLAM contraint par les plans du modèle est robuste à des imprécisions de modèle, notamment pour un modèle reconstruit par une méthode utilisant le capteur Kinect (Newcombe *et al.* (2011); Whelan *et al.* (2012)) ou des méthodes de reconstruction dense multi-vues telles que celle utilisée dans le logiciel `123D Catch` d'Autodesk (Vu *et al.* (2009)). Il s'agit d'un logiciel de reconstruction par vision utilisant une méthode de type SfM. La figure 5.15 représente en (a) le modèle 3D précis utilisé dans nos expérimentations, le modèle obtenu par le logiciel `123D Catch` en (b) et le modèle obtenu par le logiciel `ReconstructMe` avec le capteur Kinect en (c). La figure 5.16 montre que la méthode de SLAM contraint fonctionne aussi bien avec un modèle 3D précis (a) qu'avec un modèle moins précis reconstruit par vision (b) ou par Kinect (c).

En effet, l'utilisation des points de l'environnement contribue à rendre la méthode naturellement robuste à des erreurs de modèle. De plus, l'utilisation d'un estimateur robuste permet également de rendre la méthode robuste à de mauvaises associations entre les points 3D reconstruits par le SLAM et les plans du modèle. Nous présentons en annexe des résultats avec des modèles très peu précis comme les modèles de villes issues d'un SIG.

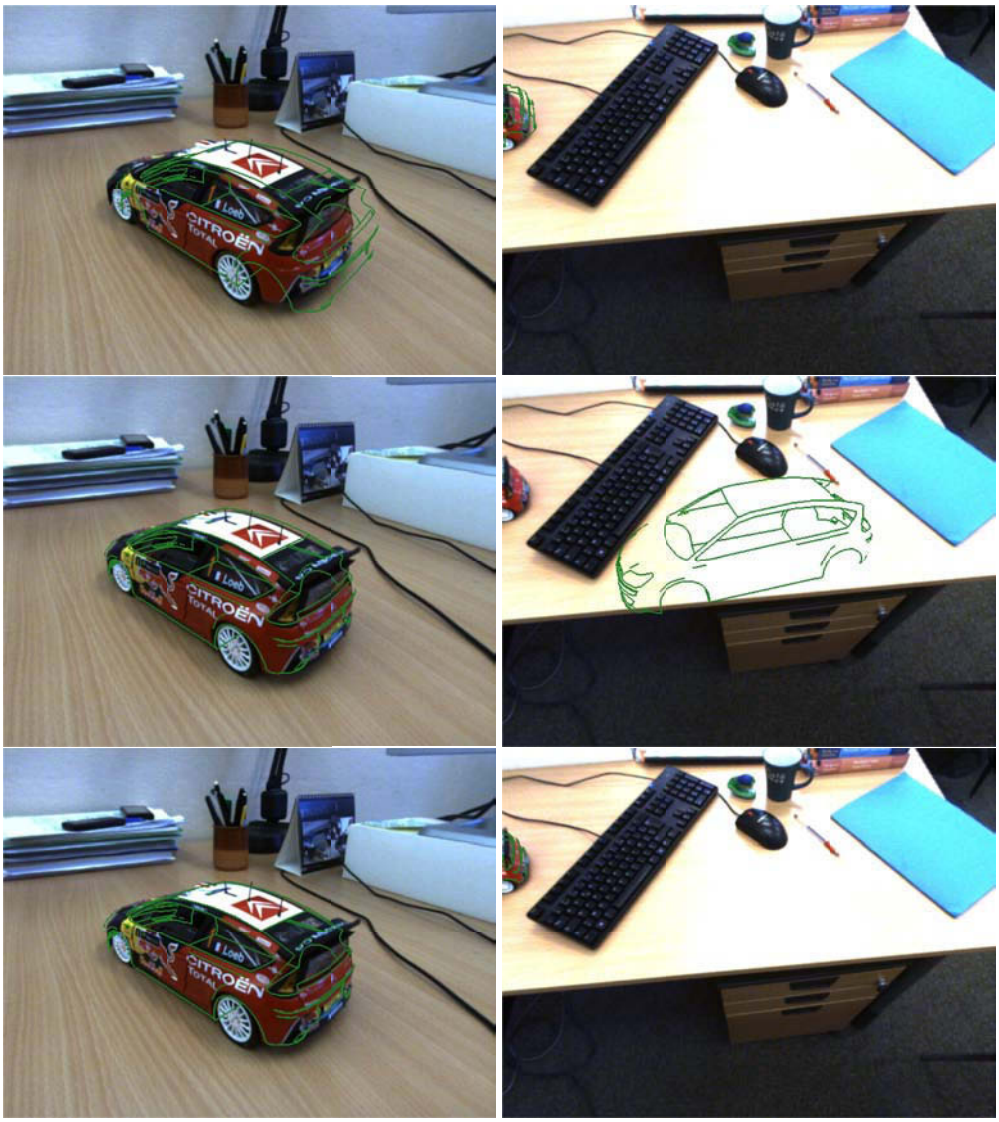


FIGURE 5.13 – Suivi d'objet 3D au moyen d'un pSLAM avec trois types d'ajustement de faisceaux local. En haut, au milieu, en bas : résultats obtenus respectivement avec les ajustments de faisceaux pBA_E, pBA_M, pBA_M&E.

5.7.2 sSLAM contraint aux plans

5.7.2.1 Evaluation sur des données simulées

Dans cette section nous comparons les trois algorithmes d'ajustement de faisceaux sBA_E, sBA_M et sBA_M&E[7] sur des données simulées. Ils minimisent

7. s signifie qu'il s'agit de primitives de type segments, M signifie que seules les contraintes au modèle (c'est-à-dire de la partie connue de l'environnement) sont utilisées tandis que M&E signifie qu'à la fois les contraintes au modèle mais aussi les informations de l'environnement sont prises en compte.

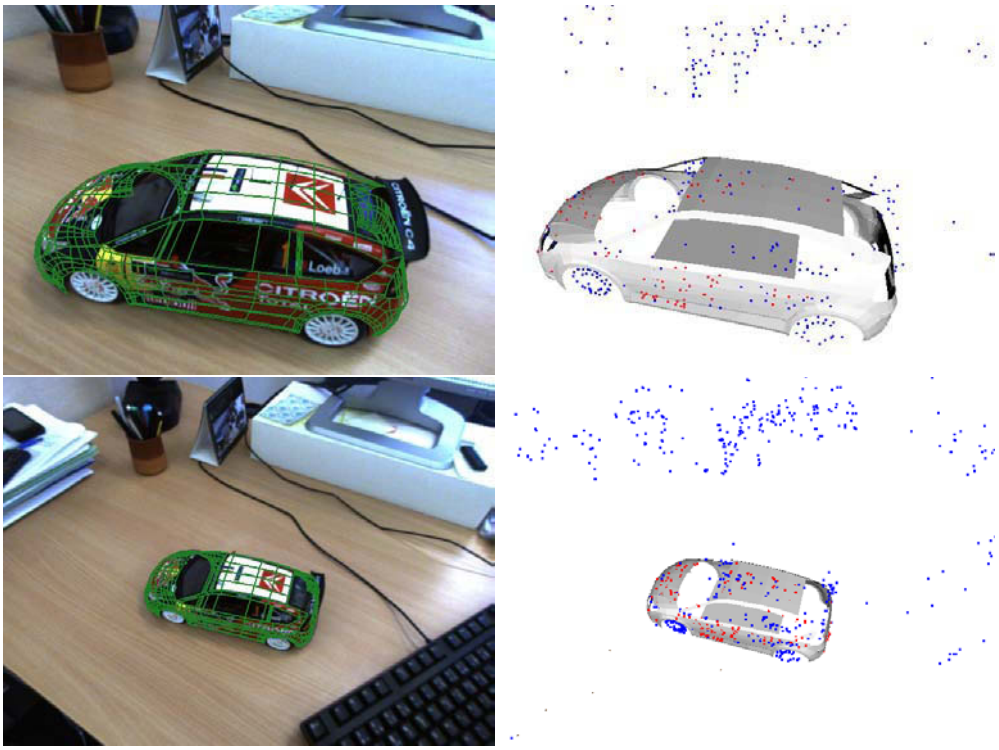


FIGURE 5.14 – Reconstruction 3D de la Citroën C4 avec le pSLAM contraint aux plans du modèle par l'algorithme pBA_M&E. En bleu, le nuage de points 3D reconstruits par le pSLAM et associé à l'environnement. En rouge, les points 3D associés aux plans du modèle.

respectivement les équations (5.9) et (5.11) en suivant la procédure décrite dans le tableau 5.3. Cette évaluation a été réalisée pour les trois algorithmes avec un ajustement de faisceaux global. La comparaison des trois algorithmes est réalisée en terme de précision de la localisation obtenue ainsi que sur la robustesse à un mauvais recalage initial.

Protocole expérimental pour la comparaison des trois algorithmes

Scène simulée. Nous simulons un jeu de caméras observant un ensemble de segments 3D, choisis au hasard dans une sphère d'un rayon de 1 mètre, par tirage aléatoire de leurs extrémités. Les caméras sont réparties autour de cette sphère sur un rayon de 2 mètres. La configuration standard consiste en 6 caméras situées à une distance de 2 mètres des segments, et une distance entre caméras consécutives (*baseline* en anglais) de 1 mètre. Le nombre de segments de l'environnement est 30, le nombre de plans est 5 et le nombre de segments sur les plans varie aléatoirement de 1 à 4. Cette scène est illustrée sur la figure 5.17.

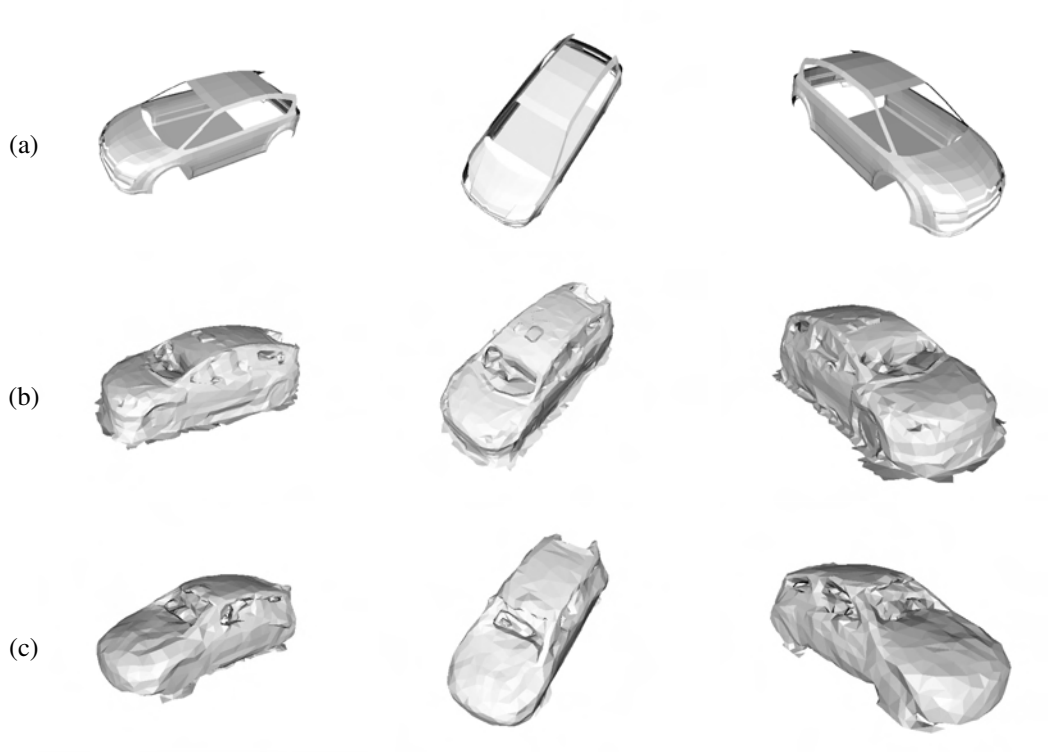


FIGURE 5.15 – Différents modèles obtenus par vision et avec la Kinect. (a) : le modèle 3D précis utilisé dans nos expérimentations est composé de 1610 plans, (b) : le modèle obtenu par reconstruction 3D avec le logiciel `123D Catch` d'Autodesk est composé de 2884 plans et (c) : le modèle obtenu par reconstruction 3D avec le logiciel `ReconstructMe` utilisant la Kinect est composé de 3316 plans.

Protocole expérimental. Nous comparons ici les trois algorithmes d'ajustement de faisceaux sBA_E, sBA_M et sBA_M&E en simulant différentes sources d'erreurs comme celle du recalage initial, de la dérive, *etc.* Trois types de perturbations sont alors générées sur la reconstruction initiale composée des poses de la caméra, des segments 3D.

- ▷ Le premier test (TEST BRUIT IMAGE) simule le bruit dans les observations 2D des segments dans les images. Les extrémités de tous les segments sont projetées dans toutes les vues et leurs projections sont bruitées avec une distribution gaussienne d'écart-type variant entre 0 et 2 pixels.
- ▷ Le deuxième test (TEST NON RIGIDE) simule les erreurs d'estimation du déplacement relatif entre deux caméras. Elles ont pour origine les algorithmes de type SLAM en présence de bruit, de valeurs aberrantes, et les dérives numériques, *etc.* Une perturbation non rigide de la reconstruction globale est réalisée en perturbant de manière aléatoire les déplacements inter-caméras. L'amplitude de perturbation varie de 10 cm et de 1 à 16 degrés.
- ▷ Le troisième test (TEST RIGIDE) simule des imprécisions du recalage initial

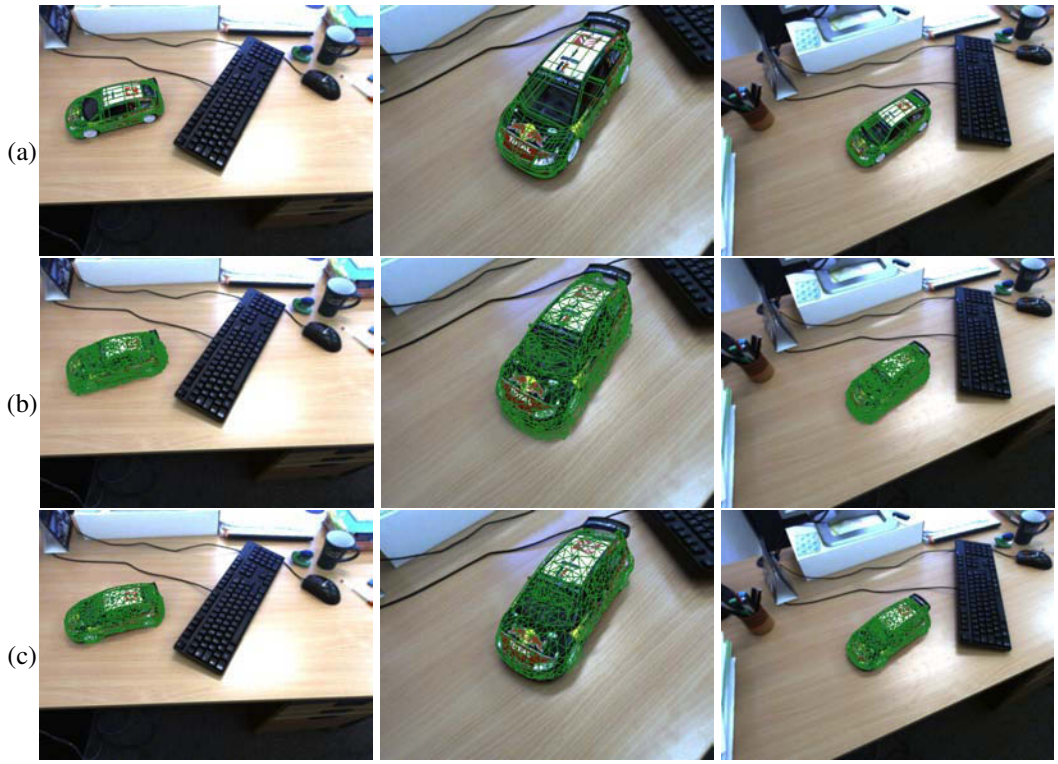


FIGURE 5.16 – Résultat de localisation obtenu avec la méthode de SLAM contraint aux plans avec différents modèles. Images pour (a) : localisation avec le modèle 3D précis, (b) : avec le modèle obtenu par vision, (c) : avec le modèle obtenu par Kinect.

entre les repères du monde et du modèle. Pour cela, une perturbation rigide est appliquée sur les poses des caméras. L'amplitude de perturbation varie de 10 cm et de 1 à 16 degrés.

Les 100 tirages aléatoires de bruit sont réalisés pour chaque amplitude. La qualité de la reconstruction finale est mesurée par le RMS 3D sur les positions et les orientations de la caméra par rapport à la vérité terrain. Les résultats sont des moyennes sur les 100 tirages aléatoires. Les trois algorithmes d'ajustement de faisceaux sont comparés en termes de précision.

Résultats

Bruit sur les observations 2D. La figure 5.18 montre les résultats obtenus lorsque les segments 2D sont bruités. sBA_M&E est l'algorithme le plus robuste aux perturbations dans les images, sBA_E a les plus mauvaises performances. Par exemple, nous observons sur la figure 5.18 que pour une amplitude de perturbation de 1 pixel pour le TEST BRUIT IMAGE, le RMS 3D de sBA_M&E est de 1 cm, celui de sBA_M est proche de 1.5 cm, alors que celui de sBA_E est de 3 cm. De manière

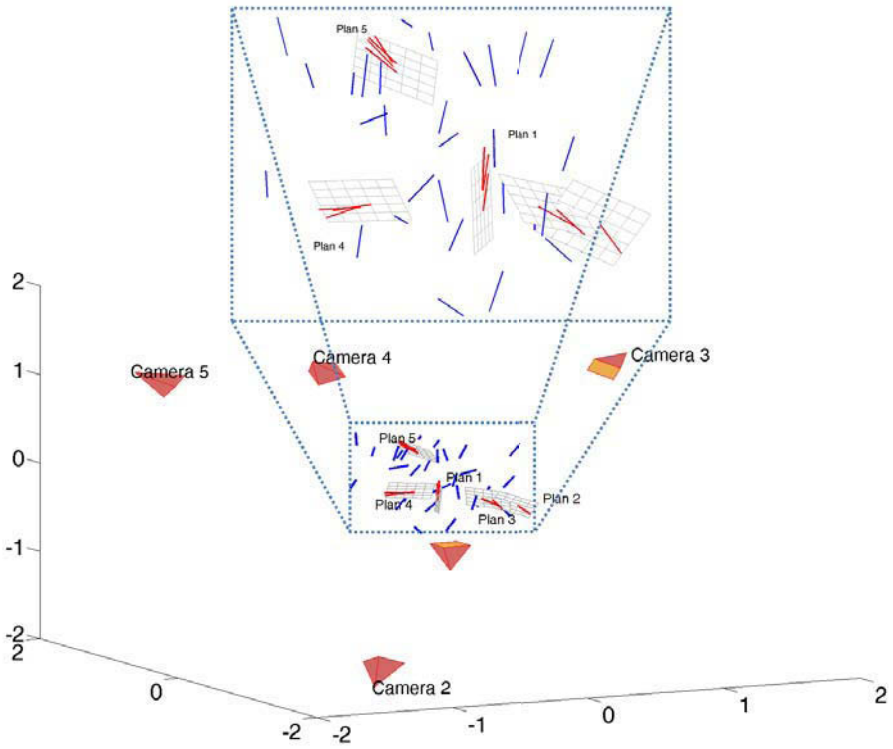
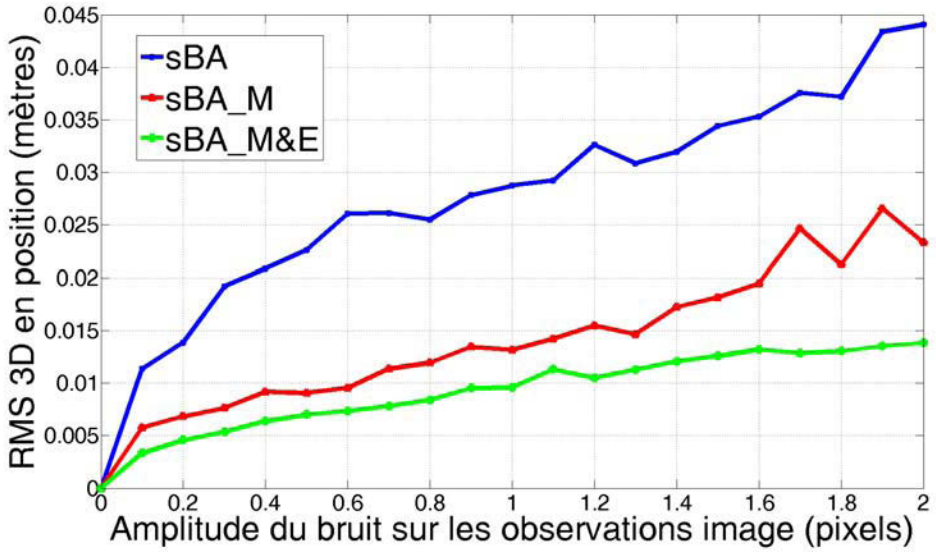


FIGURE 5.17 – La scène simulée est composée de caméra, de segments 3D ainsi que de plans 3D composés par un ensemble de segments 3D.

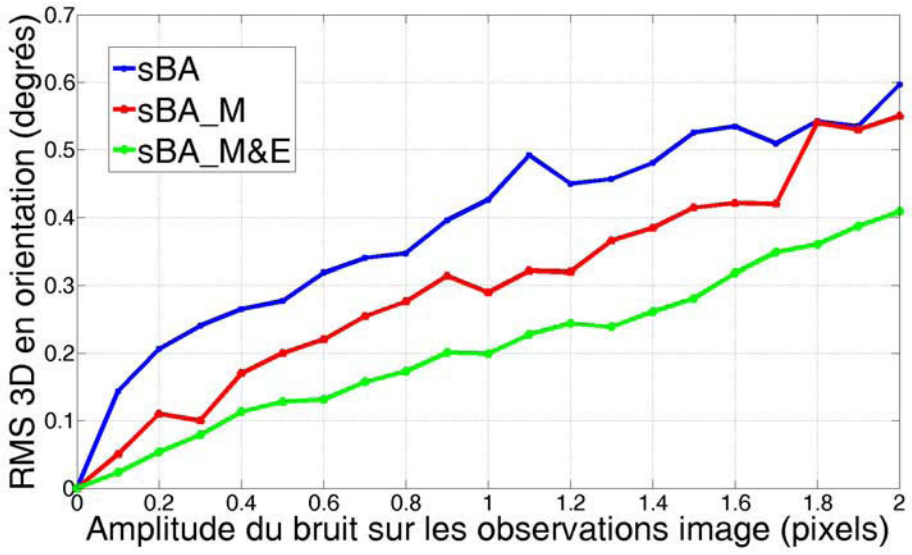
générale, nous observons que l'algorithme sBA_E donne des résultats se dégradant rapidement, alors que les résultats des autres méthodes sBA_M et sBA_M&E se dégradent plus lentement à mesure que le bruit augmente.

Bruit sur les poses des caméras. Les figures 5.19 et 5.20 montrent respectivement les résultats obtenus lorsque les positions et les orientations des caméras sont bruitées. Les poses des caméras sont bruitées indépendamment les unes par rapport aux autres, on parle alors de TEST NON RIGIDE. Pour ce test, l'algorithme sBA_M&E donne les meilleurs résultats, alors que sBA_E a les plus mauvaises performances. Par exemple, pour une amplitude de perturbation de 8 degrés (figure 5.20), pour le TEST NON RIGIDE, le RMS 3D en position de l'algorithme sBA_M&E est inférieur à 2 cm, celui de sBA_M est de 4 cm, alors que celui de sBA_E est de 15 cm. Pour la même amplitude de perturbation, le RMS 3D en orientation de l'algorithme sBA_M&E est de 1 degré, celui de sBA_M est supérieur à 2 degrés, alors que celui de sBA_E est de 7 degrés. Pour ce dernier algorithme l'erreur n'a pas beaucoup diminuée.

A noter que par soucis de clarté, les résultats pour le TEST RIGIDE ne sont



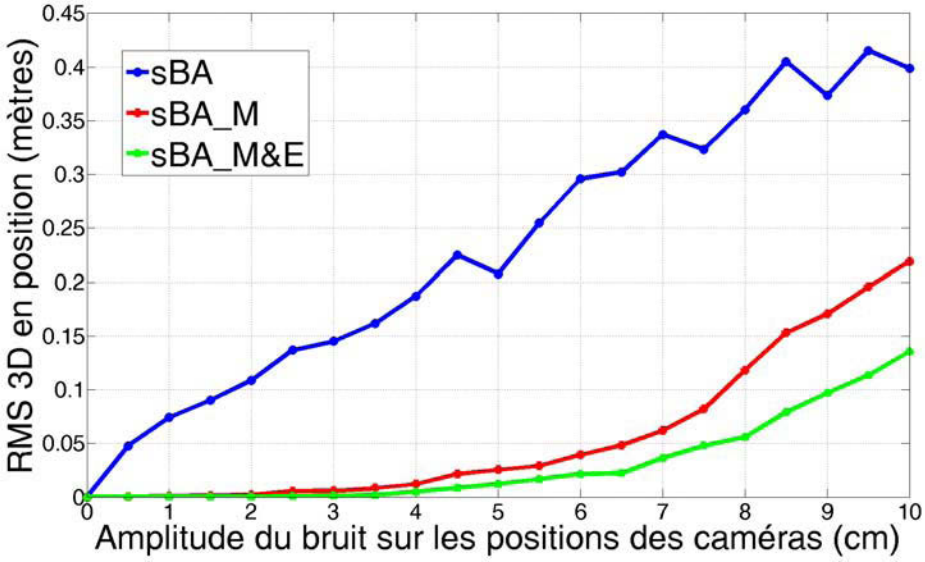
(a)



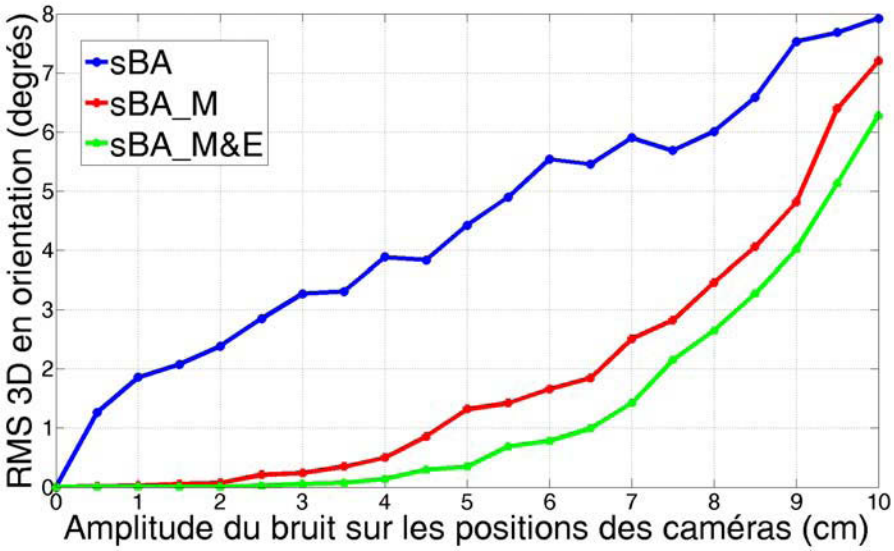
(b)

FIGURE 5.18 – **Evaluation des algorithmes sBA_E, sBA_M et sBA_M&E à la robustesse au bruit dans les observations 2D.** Résultats obtenus avec les trois algorithmes pour le TEST BRUIT IMAGE. La précision est mesurée par le RMS 3D en position (a) et en orientation (b).

pas reportés dans le mémoire car les trois algorithmes se comportent de manière similaire au TEST NON RIGIDE. L'algorithme sBA_M&E est donc plus précis que sBA_E et sBA_M pour les deux types de perturbations (TEST NON RIGIDE et TEST RIGIDE).



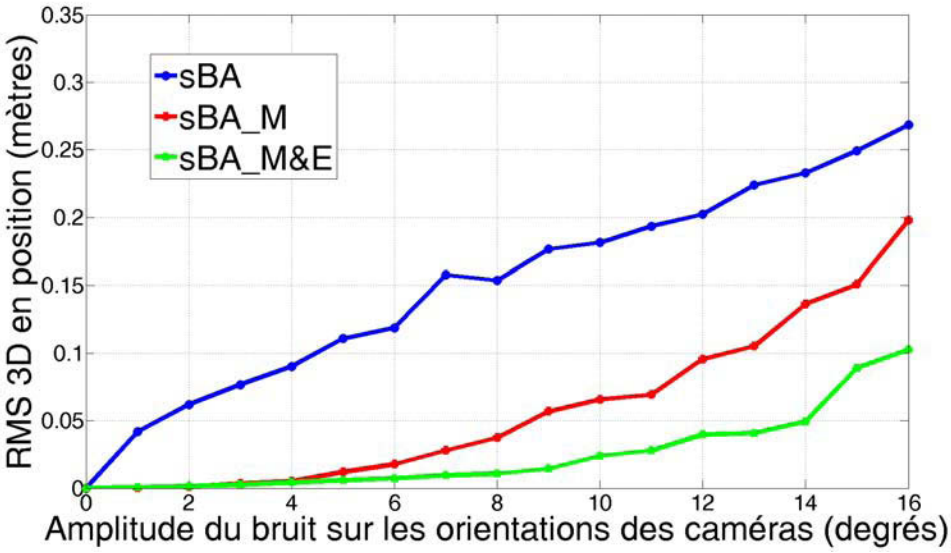
(a)



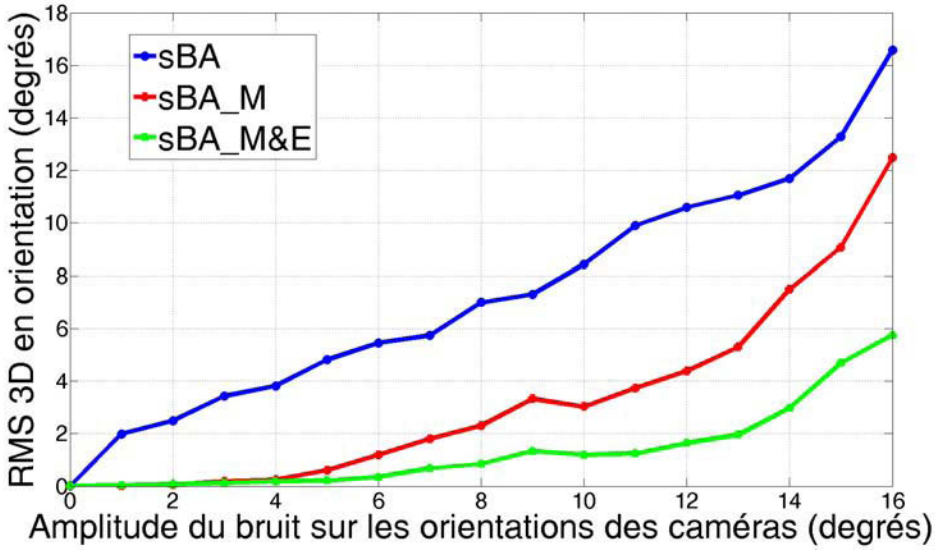
(b)

FIGURE 5.19 – **Evaluation des algorithmes sBA_E, sBA_M et sBA_M&E à la robustesse au bruit sur les positions des caméras.** Résultats obtenus avec les trois algorithmes sBA_E, sBA_M et sBA_M&E pour le TEST NON RIGIDE. La précision est mesurée par le RMS 3D en position (a) et en orientation (b).

Résumé. Ces premiers résultats obtenus en simulation montrent qu'à partir d'une reconstruction initiale imparfaite, pouvant provenir d'un sSLAM, un raffinement de cette reconstruction par un ajustement de faisceaux contraint (avec des segments



(a)



(b)

FIGURE 5.20 – **Evaluation des algorithmes sBA_E, sBA_M et sBA_M&E à la robustesse au bruit sur les orientations des caméras.** Résultats obtenus avec les trois algorithmes sBA_E, sBA_M et sBA_M&E pour le TEST NON RIGIDE. La précision est mesurée par le RMS 3D en position (a) et en orientation (b).

contraints aux plans du modèle et des segments de l'environnement) donne les meilleures performances.

Pour pousser cette étude théorique, il serait intéressant de tester celui-ci sur des données réelles. Cependant, nous n'avons pas encore développé un sSLAM

suffisamment mature pour créer de manière automatique une carte de segments 3D.

5.8 Conclusion

Nous avons présenté différents algorithmes de raffinements non linéaires d'une reconstruction de type SLAM avec mise à jour du modèle. Deux fonctions de coût, une pour les points et une pour les segments ont été proposées. Ces fonctions de coût prennent en compte à la fois l'information géométrique fournie par les plans du modèle et l'information multi-vues fournie par les primitives de l'environnement. Dans le cas du pSLAM la combinaison optimale des deux termes de la fonction de coût ainsi que le processus d'optimisation ont été étudiés. Des tests sur données de synthèse et réelles montrent l'apport de l'environnement dans l'ajustement de faisceaux. Cela permet d'augmenter la précision de la reconstruction, la robustesse et le bassin de convergence. Nous avons appliqué avec succès ce processus au suivi d'objet 3D.

Le sSLAM contraint au modèle a été présenté ici comme une extension du pSLAM contraint pour lequel nous avons décrit le formalisme théorique. Une vérification expérimentale des fondements de l'approche d'ajustement de faisceaux contraint aux plans du modèle, dans le cas de primitives segments a été réalisée sur des données simulées. Cette étude montre encore une fois l'apport des primitives de l'environnement dans l'ajustement de faisceaux.

CHAPITRE

6

Application à la réalité augmentée

Dans les chapitres précédents, nous avons présenté le principe de SLAM contraint par un modèle 3D. Nous avons vu que ce concept constitue un cadre générique et flexible pour la localisation d'une caméra. Dans ce chapitre nous validons cette solution sur différentes applications de réalité augmentée dédiées d'une part au domaine de l'automobile : pour la maintenance assistée d'un moteur de voiture, la personnalisation de la carrosserie et un guide d'utilisation pour le tableau de bord. D'autre part nous présentons une application de design de cuisine ainsi qu'un prototype de réalité augmentée pour des applications de formation en milieux industriels. Les résultats obtenus ont permis de développer des partenariats industriels dans le domaine de la RA et ont mené à deux publications (Besbes et al. (2012); Gay-Bellile et al. (2012)). Ces travaux d'intégration ont fait l'objet d'un travail d'équipe au sein du laboratoire.

6.1 Milieu automobile

6.1.1 Introduction

La réalité augmentée a assurément un grand potentiel dans le domaine de l'automobile car il y a de nombreuses applications dans lesquelles l'ajout d'informations virtuelles a un réel intérêt. L'efficacité des processus de vente et des services d'après-vente dans le domaine de l'automobile peut être grandement améliorée en utilisant la technologie de réalité augmentée, notamment par la personnalisation par le client des voitures en concession. De plus, la réalité augmentée peut permettre d'expliquer le fonctionnement de certains éléments du véhicule ce qui permet à l'utilisateur une compréhension et une appropriation plus rapide de ces éléments.

Bien qu'il existe un grand nombre d'applications potentielles, les systèmes de réalité augmentée dans le domaine de l'automobile ne sont pas très répandus. La raison principale est que ces applications nécessitent une localisation de la caméra très précise et stable pour obtenir un rendu réaliste. Ceci est très important pour l'acceptabilité des systèmes de réalité augmentée dans ce domaine. Jusqu'à présent, les solutions de localisation existantes ne parviennent pas à répondre à ces deux exigences.

Dans les chapitres précédents nous avons démontré que la solution de SLAM contraint répond à ces exigences de précision et de stabilité. Ces résultats ont mené à une étude collaborative entre Renault, DiotaSoft et le CEA LIST. L'objectif de cette étude est d'évaluer la capacité de cette méthode de localisation à fournir une solution pour une plate-forme de réalité augmentée dédiée au secteur automobile. Trois scénarios de RA sont considérés pour cette étude : la personnalisation interactive de la carrosserie de la voiture, un guide d'utilisation pour les éléments du tableau de bord et la maintenance assistée pour le moteur de la voiture. Ces scénarios permettent d'évaluer les performances de la localisation pour les différentes parties de la voiture (tableau de bord, carrosserie, moteur). Ils illustrent également, grâce à des applications réelles, les avantages de la RA pour l'industrie automobile.

La section 6.1.2 décrit brièvement les travaux liés à la réalité augmentée dans le domaine de l'automobile. Ensuite, nous présentons notre plate-forme de RA s'appuyant sur une tablette grand public incorporée dans une tablette Selltic [1] destinée à améliorer l'expérience de RA. Nous présentons les résultats de l'évaluation de la localisation pour les trois scénarios dans la section 6.1.3.

6.1.2 Travaux de réalité augmentée liés au domaine de l'automobile

L'automobile est sans doute le domaine qui est le plus fréquemment utilisé pour illustrer le potentiel de la réalité augmentée. Les premières applications basées sur une localisation par marqueurs ont été utilisées pour la visualisation de voitures virtuelles en ligne, voir ARV; Kalkofen *et al.* (2007). Ensuite, l'amélioration de la localisation par les méthodes basées sur le suivi de patch, par exemple Taylor *et al.* (2009) ont permis le déploiement de la publicité virtuelle par RA sur des planches publicitaires, et tout particulièrement pour la publicité dans le domaine de l'automobile [2]. Les voitures miniatures sont aussi fréquemment utilisées pour démontrer la capacité des systèmes de suivi basé modèle, voir par exemple Wuest *et al.* (2007b).

Néanmoins, malgré que la vidéo de concept [3] proposée par BMW pour l'aide

1. Tablette de réalité augmentée conçue par DiotaSoft.
2. `http://www.youtube.com/watch?v=pFS6EHzBGVc`
3. `http://www.bmw.com/com/en/owners/service/augmented_reality_introduction_1.html`

à la maintenance soit l'une des plus célèbres vidéos de réalité augmentée, les systèmes de RA pour les voitures de taille réelle ne sont pas très répandus. Cela est dû au fait que la voiture n'est pas favorable aux solutions de suivi, qu'elles soient basées sur l'apparence ou sur la géométrie. A notre connaissance, seul Metaio a démontré dans Platonov *et al.* (2006), la capacité de leur système de localisation 3D par vision à gérer le cas d'une voiture réelle. Néanmoins, étant donné que leur processus de localisation utilise exclusivement l'apparence de l'objet, cette solution n'a été démontrée que sur la partie la plus texturée de la voiture, c'est-à-dire le moteur.

Parmi les éléments d'une voiture, la carrosserie est l'élément le plus difficile à gérer pour les méthodes de suivi. D'une part, en raison de son grand volume, la carrosserie ne peut pas être entièrement visible depuis un seul point de vue. Ajouté à cela, il y une distribution inégale des contours francs, ce qui fait que les méthodes de suivi utilisant les contours du modèle, telles que Drummond et Cipolla (2002); Lepetit et Fua (2005); Wuest *et al.* (2007b), sont instables. D'autre part, le manque de texture sur la carrosserie du véhicule et la présence des spécularités et des réflexions ne sont pas favorables pour les méthodes de suivi utilisant l'apparence, telles que Lepetit et Fua (2005); Skrypnyk et Lowe (2004); Vacchetti et Lepetit (2004). Les solutions existantes pour les parties peu texturées d'une voiture utilisent des capteurs supplémentaires très couteux, tels qu'un bras robotisé, avec une haute précision de mesure, ou encore l'utilisation d'un système de localisation ART. C'est pourquoi la plupart des démonstrations de réalité augmentée dans les derniers salons automobiles (par exemple, Mahindra XUV500 à l'Auto Expo 2012 [4]) reposent toujours sur des technologies de localisation d'une caméra fixe.

Dans la section 6.1.3 nous validons que notre solution permet de proposer une localisation par rapport aux différents éléments d'une voiture (carrosserie inclue), pour des applications de réalité augmentée mobiles.

6.1.3 Résultats expérimentaux et applications à différents scénarios de RA

Dans cette section nous démontrons que notre solution de localisation, décrite dans les chapitres précédents, offre une flexibilité, une précision et des performances suffisantes pour gérer les principales applications de RA dans le domaine automobile. Tout d'abord, nous décrivons la plateforme mobile utilisée dans les expérimentations. Ensuite, trois applications sont considérées et chacune d'entre elles se concentre sur une partie spécifique de la voiture : le moteur, la carrosserie de la voiture et le tableau de bord.

4. `http://www.youtube.com/watch?v=iA-2ElU9Qs8`

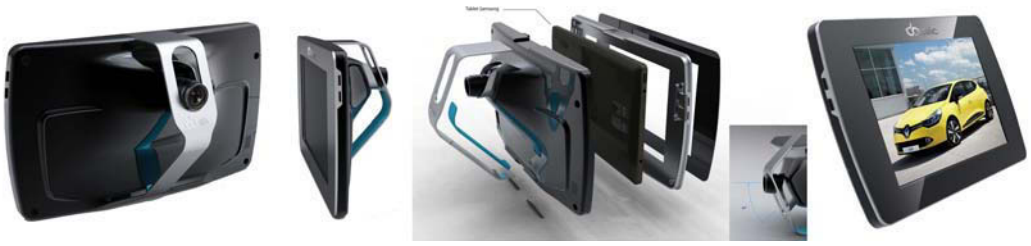


FIGURE 6.1 – La tablette Selltic, comprenant une tablette PC Samsung et une caméra uEye.

Configuration de la plateforme mobile

Notre système de réalité augmentée utilise une tablette Selltic [5], qui a été conçue à l'origine comme une solution technologique pour le domaine de la vente consultative. Le boîtier de la tablette a été spécialement conçu pour accueillir tous les composants matériels du système et par la même occasion pour fournir une meilleure ergonomie pour l'utilisateur. La tablette Selltic utilise une tablette Samsung Series 7 (avec un processeur Intel Core i5-2467M , une carte graphique Intel HD 3000). Une caméra grand angle uEye 2210SE (de longueur focale 3.5 mm, 640×480 pixels) fournissant jusqu'à 45 images par seconde, est intégrée dans la coque de la tablette Selltic, comme présenté sur la figure 6.1. Le signal vidéo de la caméra est transmis à la tablette PC, au sein de laquelle la localisation de la caméra et le rendu sont réalisés. Notez que notre approche de localisation peut fonctionner directement à l'aide de la caméra intégrée de la tablette. Diotasoft a décidé d'utiliser une caméra externe pour avoir une meilleure qualité d'image et ainsi améliorer l'expérience de réalité augmentée de l'utilisateur. Le temps de traitement de l'algorithme de localisation sur cette tablette est de 25 ms en moyenne par image, ce qui est proche de la fréquence d'acquisition de la caméra.

L'entretien et la maintenance d'un moteur de voiture

La première application concerne l'assistance d'un opérateur pour l'entretien et les opérations de maintenance du moteur. Des éléments virtuels sont ajoutés afin de guider l'opérateur lors de la procédure de maintenance. Ce scénario implique que le processus de localisation soit robuste à de forts changements de point de vue, à des mouvements rapides de la caméra et à des occultations de la scène (par exemple un opérateur qui introduit sa main dans le champ de vue de la caméra).

L'objet d'intérêt est le moteur d'une Renault Clio. La voiture est placée dans un contexte extérieur ce qui implique des conditions d'éclairage non maitrisées. Ne disposant pas de modèle géométrique du moteur, nous utilisons comme modèle, un

5. $Selltic^{TM}$ est une marque distribuée par DiotaSoft qui fournit un ensemble de solutions technologiques destinées à soutenir les ventes en magasins et au-delà (`www.selltic.eu`).

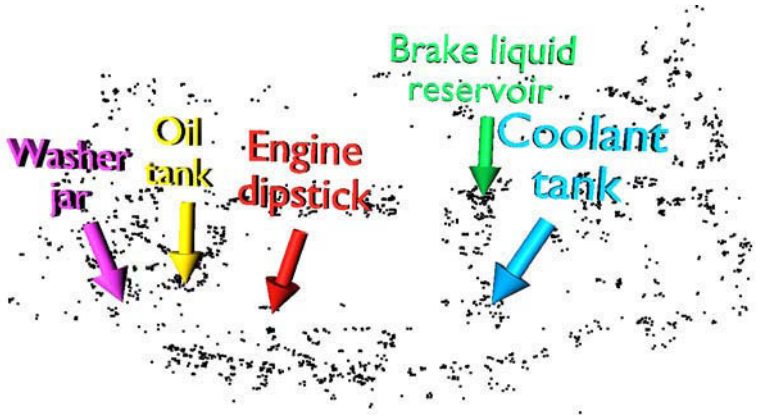


FIGURE 6.2 – Le modèle 3D du moteur sous forme de nuage de points 3D avec les informations virtuelles pré-enregistrées et utilisées pour l'augmentation.

nuage de points 3D pour contraindre l'ajustement de faisceaux (pour plus de détails voir la section 4.3). Ce modèle composé d'environ 2000 points, illustré sur la figure 6.2, est construit hors ligne avec un processus de reconstruction multi-vues.

La précision et la robustesse de la méthode de localisation peuvent être appréciées sur la figure 6.3, où les différentes parties du moteur (jauge d'huile moteur, réservoir de liquide de frein, liquide de refroidissement, *etc.*) sont mises en évidence par le système pour différents points de vue. Le système fournit une localisation précise, stable et robuste et est donc parfaitement adapté pour cette application de réalité augmentée.

Personnalisation virtuelle de la carrosserie d'une voiture

La seconde application de réalité augmentée est la personnalisation interactive de la carrosserie d'une voiture ainsi que le *tuning* virtuel. L'objectif est de permettre au client en concession automobile de voir les différents modèles disponibles d'une voiture, par exemple les couleurs et les matériaux, tout en se déplaçant autour de celle-ci. Ce scénario implique que la méthode de localisation soit robuste à l'absence de texture sur la carrosserie de la voiture. De plus, le fait que la carrosserie ne soit pas entièrement visible en permanence rend la localisation difficile, surtout si l'utilisateur s'approche de la voiture.

Comme la carrosserie de la voiture est peu texturée, la méthode d'unification SLAM et un suivi de modèle géométrique est sélectionnée (pour plus de détails voir la section 4.2). Pour obtenir des performances temps réel, le modèle CAO du véhicule utilisé dans nos expériences est simplifié à 50000 facettes. 2000 segments 3D sont ensuite extraits de ce modèle, comme illustré sur la figure 6.4. A noter que les tests ont été réalisés avec le véhicule, placé dans des parkings en intérieur et en extérieur (avec des conditions d'éclairage non maitrisées).

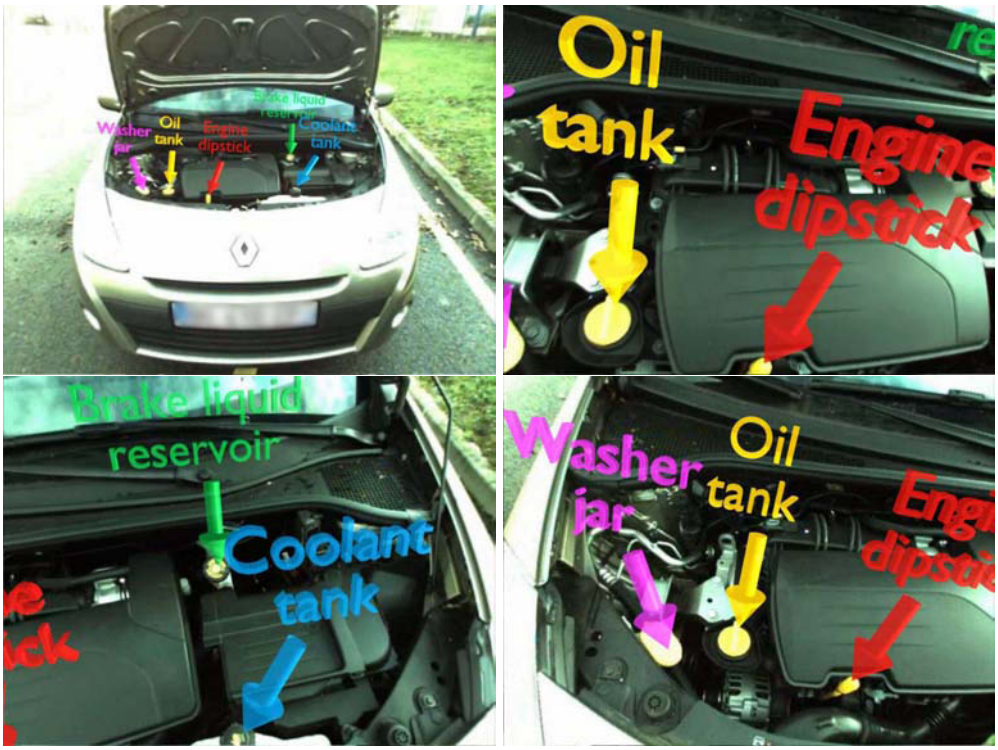


FIGURE 6.3 – Réalité augmentée sur un moteur de voiture : la position et la fonction des éléments d'entretien du moteur sont désignées virtuellement.

FIGURE 6.4 – Les segments 3D extraits du modèle CAO de la carrosserie de la voiture.

Dans ces expérimentations, la personnalisation de la voiture se fait en changeant, de façon très réaliste, la couleur de la carrosserie et en y ajoutant des bandes blanches et des éléments en plastique noir. Ainsi, la Renault Clio classique est virtuellement transformée en Clio série limité Gordini, comme illustré sur la figure 6.5. Notez que cette application, initialement conçue pour la personnalisation en concession peut être étendue au *tuning* de véhicule pour les particuliers. En effet, notre système de localisation est suffisamment précis pour permettre l'ajout d'éléments virtuels comme un aileron, pour changer la couleur des fenêtres, du pare-brise ou encore changer les pare-chocs et les jantes de la voiture. Pour cela une expérimentation a été réalisée sur un autre véhicule (voir la figure 6.6).

Personnalisation virtuelle du tableau de bord d'une voiture

La dernière application pour l'automobile concerne l'intérieur d'une voiture. La réalité augmentée est ici utilisée pour personnaliser le tableau de bord et fournir un guide d'utilisation interactif pour présenter les propriétés de la voiture en indiquant les différentes fonctionnalités des éléments du tableau de bord. Cette application est difficile à mettre en œuvre car elle nécessite une localisation très précise, malgré le manque de recul et le manque de lumière à l'intérieur du véhicule qui ne sont pas favorables aux méthodes de localisation par vision.

Comme le tableau de bord n'est pas très texturé et qu'un modèle CAO est disponible, l'unification SLAM avec un suivi basé modèle géométrique (voir la section 4.2) est sélectionnée. De même que précédemment, le modèle 3D simplifié du tableau de bord est composé de 50000 triangles à partir duquel nous avons extrait 2000 segments 3D.

Les résultats obtenus montrent que la localisation est précise tout au long de la séquence en dépit des grandes variations d'échelle de la scène et des déplacements rapides de la caméra, comme illustré sur la figure 6.7. Le *design* du tableau de bord

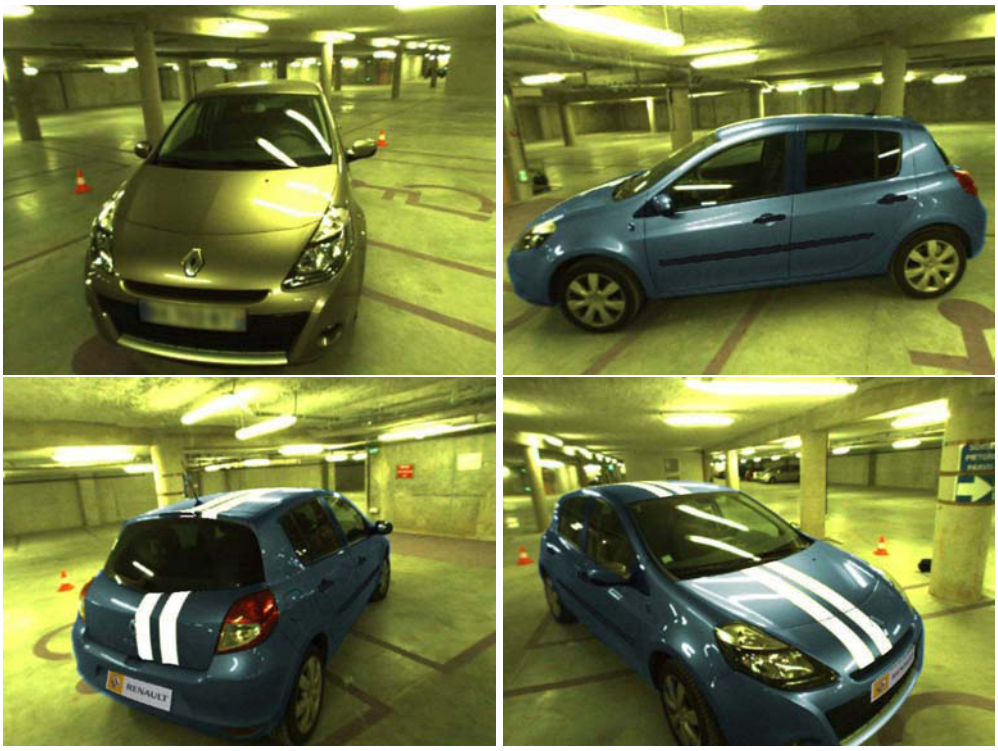


FIGURE 6.5 – Réalité augmentée sur la carrosserie d'une voiture réelle. La voiture avec sa couleur d'origine (première ligne à gauche) et la même voiture personnalisée virtuellement. Notons que pour une meilleure appréciation de la précision de la méthode seule la carrosserie a été modifiée, les roues, les fenêtres et le pare-brise ne sont pas augmentés.

FIGURE 6.6 – Réalité augmentée sur la carrosserie d'une voiture réelle dans un contexte extérieur. La voiture avec sa couleur d'origine (en haut à gauche) et la même voiture « tunée ». Nous avons modifié certains éléments comme les roues et les plaques d'immatriculation, nous avons également ajouté un aileron, et changé la couleur des fenêtres, du pare-brise et des pare-chocs de la voiture.

FIGURE 6.7 – Personnalisation virtuelle, par réalité augmentée, du tableau de bord d'une voiture. Le tableau de bord d'origine de la voiture (en haut à gauche) est personnalisé. Un GPS virtuel est ajouté et la couleur de la baguette au dessus de la boite à gant est changée en doré (à droite). Des informations virtuelles sont également ajoutées pour présenter certains éléments du tableau de bord (en bas à gauche).

de la voiture a été changé et les fonctionnalités de certains éléments sont indiquées virtuellement.

6.2 Personnalisation virtuelle d'une cuisine

Cette application de réalité augmentée est la personnalisation interactive d'une cuisine. L'objectif est de permettre au client en magasin de voir les différents modèles disponibles d'une cuisine, par exemple les couleurs, les matériaux, les formes des poignées, tout en se déplaçant autour de celle-ci. Ce scénario implique que la méthode de localisation soit robuste à l'absence de texture sur la cuisine.

Comme la cuisine est peu texturée, l'unification SLAM et un modèle géométrique est sélectionnée (pour plus de détails voir la section 4.2). Pour obtenir des performances temps réel, le modèle CAO de la cuisine est simplifié à 50000 facettes. 2000 segments 3D sont ensuite extraits de ce modèle, comme illustré sur la figure 6.8. La reconstruction 3D obtenue de la cuisine et de son environnement,

est illustrée sur la figure 6.9. Le nuage de points 3D reconstruits par le SLAM est représenté en bleu et les segments 3D extraits du modèle sont représentés par des segments bleus avec leurs points milieu en rouge. La trajectoire de la caméra est représentée par des triangles bleus.

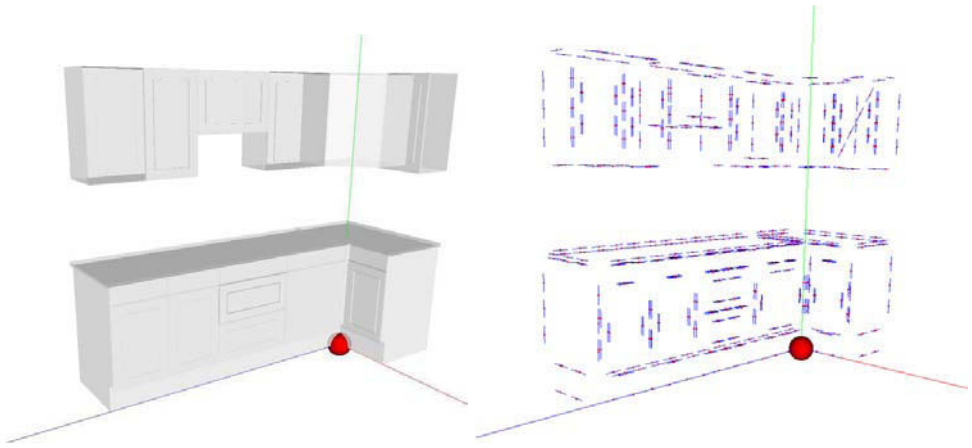


FIGURE 6.8 – Les segments 3D extraits du modèle de la cuisine.

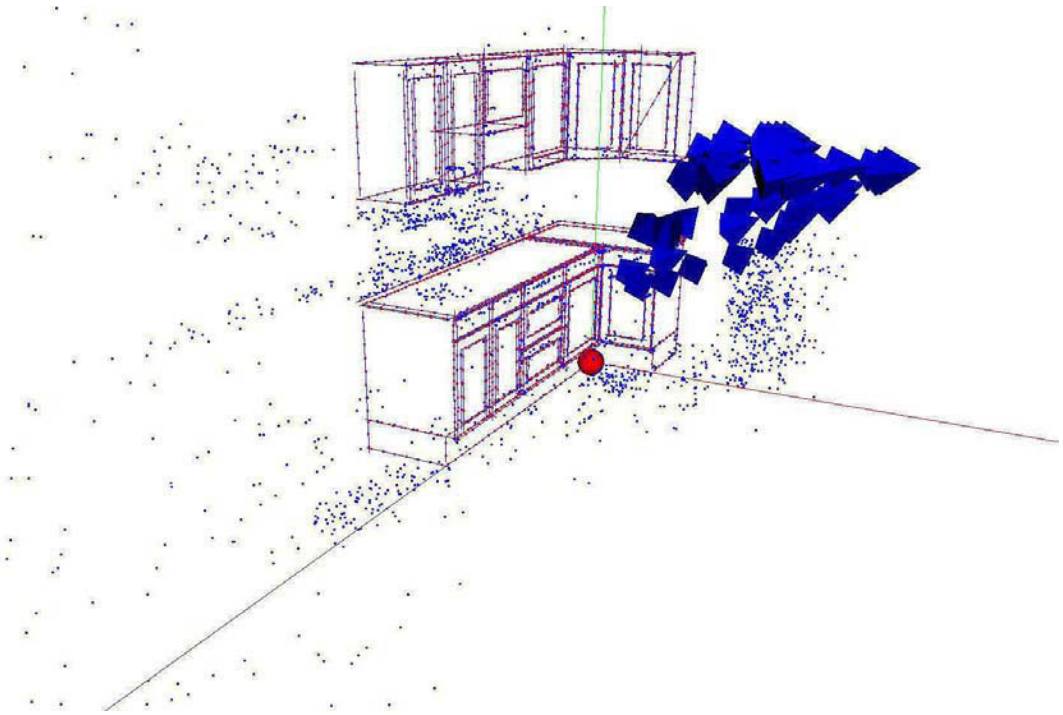


FIGURE 6.9 – Reconstruction 3D de la cuisine. En bleu, le nuage de points 3D reconstruits par le SLAM et en rouge, les segments 3D extraits du modèle. Les cones bleus représentent les caméras.

Dans ces expérimentations, la personnalisation de la cuisine se fait en changeant, de façon très réaliste, chaque partie de la cuisine. Par exemple, le revêtement, le mobilier, le plan de travail, peuvent être modifiés séparément ou simultanément. Lorsque l'utilisateur clique sur un élément de la cuisine, un menu apparait dans les images. Il contient les différents *designs* disponibles en magasin pour l'élément en question. Une fois que le *design* est choisi, la cuisine est automatiquement augmentée avec celui-ci en temps réel. Un rendu réaliste est obtenu en modélisant les sources lumineuses de la salle. La figure 6.10 représente les différentes fonctionnalités du logiciel de *design* de cuisine. Notons qu'il est également possible d'ajouter du mobilier ou éventuellement d'en remplacer. Cela est illustré par la substitution d'un meuble de la cuisine par une hotte.

La réalisation de l'application de *design* de cuisine a été faite en collaboration avec la société Diotasoft. A noter que comme pour les applications de réalité augmentée pour l'automobile présentées précédemment, la tablette Selltic a également été utilisée pour démontrer les performances de la méthode proposée.

FIGURE 6.10 – Application de *design* de cuisine par réalité augmentée. La cuisine originale (en haut à gauche) est modifiée, en ligne, de manière virtuelle avec un autre plan de travail, une autre crédence, d'autres couleurs pour le mobilier et avec l'ajout ou le remplacement de mobilier. Un menu est proposé pour chaque composant de la cuisine afin de choisir le *design* à lui appliquer.

6.3 Aide à la formation industrielle

L'un des facteurs clés dans l'intégration de la réalité augmentée dans le domaine industriel, est de fournir une application d'assistance aux procédures de maintenance. Lorsqu'il est confronté à un objet complexe, l'utilisateur prend beaucoup de temps à se familiariser avec les aspects mécaniques ou électriques des différents composants, et différentes opérations et procédures de cet objet. Dans les recherches précédentes Farkhatdinov et Ryu (2009); Henderson et Feiner (2009), il a été démontré que la RA est un bon outil pour faciliter la compréhension et l'exécution des tâches industrielles. Afin de bénéficier de ces fonctionnalités, le système de RA doit être ergonomique. Il doit notamment permettre une interaction intuitive.

6.3.1 Description du démonstrateur

Dans cette section nous proposons un système de RA par vision directe composé d'un casque semi-transparent dans lequel les informations virtuelles viennent s'afficher (OST pour *Optical see-through*). Le prototype proposé permet de plus des intéractions dynamiques : des informations parfaitement recalées sur la scène réelle, s'affichent en réponse aux demandes de l'utilisateur. Contrairement aux systèmes de localisation existants en maintenance industrielle, nécessitant l'installation de matériels spécifiques tels que des marqueurs, des émetteurs ou des capteurs de positionnement, ce système n'utilise que les images fournies par la caméra intégrée dans le casque (HMD pour *Head-mounted display*) et un modèle de l'objet d'intérêt. La formation s'appuie sur les interactions de l'utilisateur qui se font simplement en indiquant un élément spécifique de l'objet à l'aide d'un pointeur laser.

Le système comporte deux modules de vision : la localisation de la caméra et une gestion des interactions de l'utilisateur.

6.3.2 Localisation de la caméra

Une augmentation interactive d'un objet ou d'un de ses composants, nécessite une localisation en temps réel, stable et précise, du point de vue de l'utilisateur. Une fois que la tâche de localisation de la caméra est réalisée avec précision, les animations virtuelles pour la formation peuvent être correctement positionnées et orientées par rapport au point de vue de l'utilisateur.

Les performances du suivi d'un objet industriel métallique sont fortement affectées par les reflets spéculaires et par le fait que ses surfaces ne soient pas texturées. Afin de remédier à ces problèmes, nous avons utilisé la méthode de localisation proposée en section 4.2 du chapitre 4, unifiant un SLAM avec un suivi basé modèle géométrique.

Cette solution montre une grande fiabilité sur des pièces industrielles car elle permet de gérer en temps réel, avec précision et sans *jittering*, la localisation de la caméra et ceci avec une robustesse aux occultations et aux spécularités.

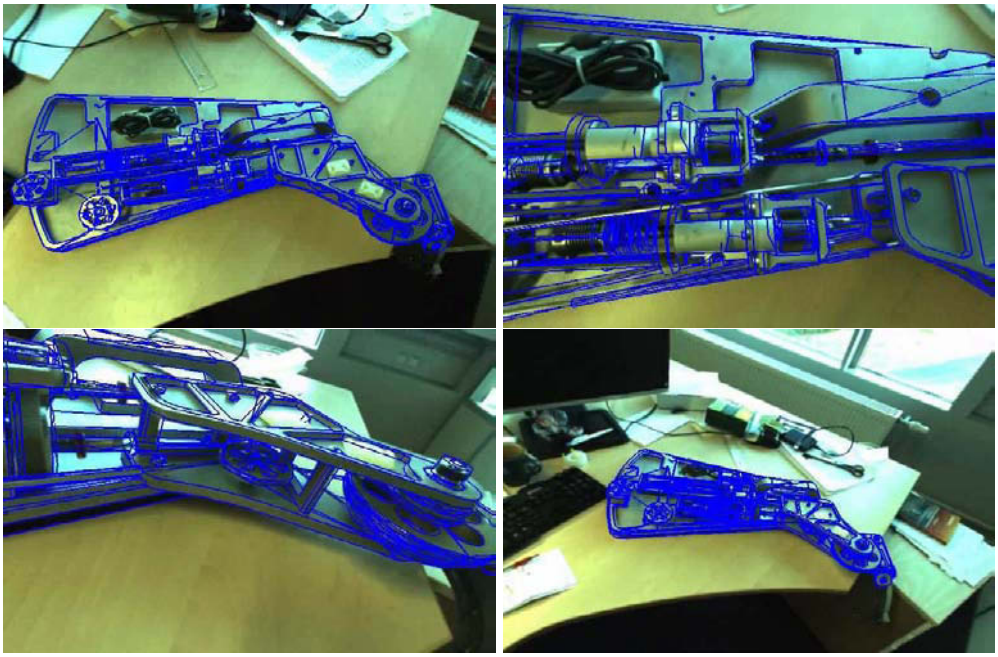


FIGURE 6.11 – Résultats de recalage du modèle 3D sur une pièce industrielle avec le pSLAM contraint par segments 3D du modèle. La pièce correspond au dos du membre supérieur d'un exosquelette.

6.3.3 Gestion des intéractions utilisateur

Pour le domaine industriel, un système de RA doit être ergonomique et doit notamment permettre une interaction intuitive. Nous proposons donc une solution d'intéraction à distance à l'aide d'un pointeur laser. La problématique consiste alors à détecter un pointeur laser sur les surfaces spéculaires d'un objet à l'aide d'un OST HMD portable équipé d'une caméra miniature de faible qualité.

L'interaction avec un pointeur laser est une technique rapide et précise pour interagir à distance. Dans ce prototype, c'est la même caméra (montée sur le casque HMD) qui est utilisée pour le suivi du point de vue de l'utilisateur, et pour la détection du pointeur laser. Une fois la pose 3D de la caméra estimée, une région d'intérêt (ROI) est définie dans l'image courante. Ensuite, le point rouge du laser est localisé dans la ROI. Pour savoir quel élément de l'objet est désigné par l'utilisateur, nous utilisons les techniques de lancer de rayons. Connaissant le modèle CAO de l'objet, il suffit alors de déterminer quel élément est intersecté par le pointeur laser. Une étape de vérification spatio-temporelle est inclue afin de confirmer la demande de

l'utilisateur. Lorsqu'un élément de l'objet est ainsi sélectionné, le système superpose les instructions pré-enregistrées concernant cet élément.
Pour la plupart des applications, le centre du point du laser est détecté à partir de la moyenne pondérée des pixels brillants de l'image après avoir réalisé un simple seuillage. Comme notre objet d'intérêt est soumis à des réflexions spéculaires, cette méthode ne peut pas être appliquée de manière fiable. Une autre façon de traiter le problème ci-dessus est de détecter et masquer, lors des premières images, les zones de fortes intensités et de couleur rouge dans lesquelles la détection n'est pas possible. Dans notre module de détection, un filtre rouge, dans le modèle de couleur HSV est d'abord appliqué. Ensuite, les pixels isolés sont enlevés par une simple opération morphologique. Enfin, le point du laser est détecté en appliquant un algorithme de détection de *blob*. A noter que nous avons appliqué le même algorithme pour détecter le doigt de la main de l'utilisateur, en tant que mode d'interaction alternatif au laser. Les résultats expérimentaux sont très prometteurs, mais moins efficaces que l'interaction basée sur le laser. La principale limitation de cette interaction alternative basée sur le doigt, se produit lorsque l'utilisateur pointe avec plus d'un doigt. Il est alors difficile de distinguer le doigt réellement utilisé par l'utilisateur pour interagir avec le système des autres doigts visibles dans l'image courante.

6.3.4 Résultats

Le prototype intègre un casque semi-transparent de Laster Technologie équipé d'une caméra embarquée, d'un pointeur laser ordinaire et d'un PC portable (Quad-Core Xeon X5482 3,2 GHz, 3,25 Go de RAM). L'OST HMD permet d'afficher des informations virtuelles dans le champ de vision de l'utilisateur.

Le système de formation guide, étape par étape, un opérateur à travers une procédure d'assemblage ou de démontage pour un objet d'intérêt spécifique. L'utilisateur peut interagir à distance avec le système, en sélectionnant simplement un élément spécifique de l'objet à l'aide d'un pointeur laser ordinaire. Comme le montre la figure 6.12, le système est capable d'augmenter, de manière interactive l'élément sélectionné, par une animation virtuelle de montage ou de démontage obtenue grâce au modèle CAO.

Le premier module comprend la méthode de localisation de caméra décrite dans ce mémoire, qui fonctionne également sur des objets industriels, complexes, métalliques présentant de nombreuses spécularités. Dans le deuxième module, nous avons développé des méthodes rapides de traitements d'image pour le suivi du point rouge d'un pointeur laser. Grâce à ces traitements, le système proposé est en mesure d'augmenter de manière interactive et en temps réel un objet industriel afin de rendre le processus d'apprentissage, plus attractif et intuitif. Ce dispositif est particulièrement utile pour sélectionner, à distance, de manière rapide, et avec précision un composant particulier d'un objet.

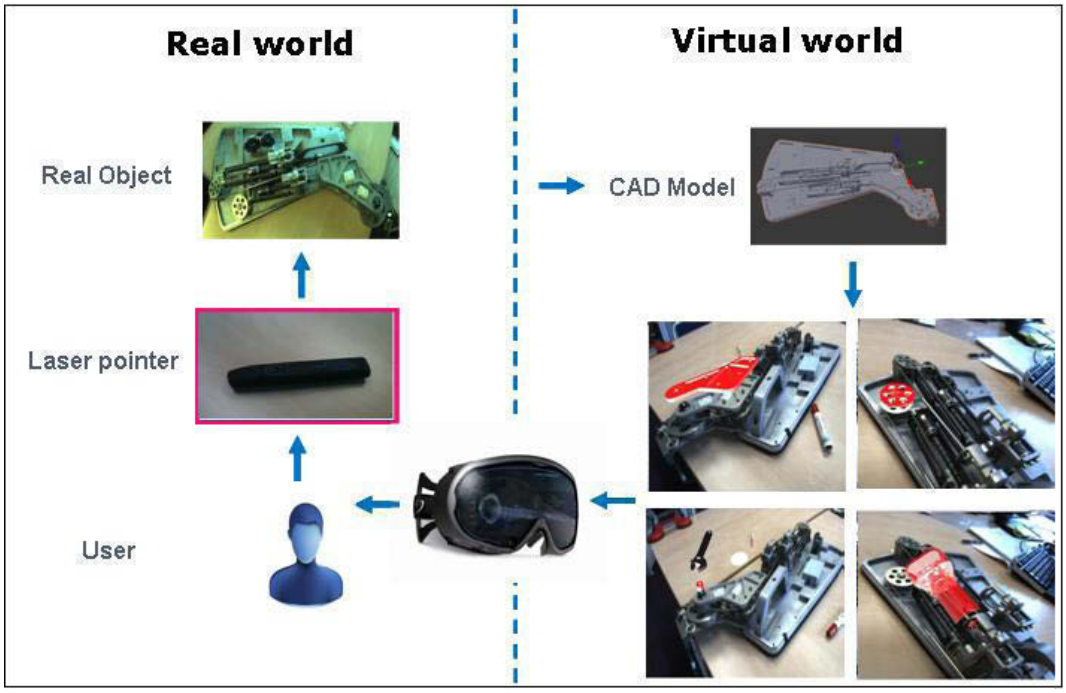


FIGURE 6.12 – La description de l'architecture du prototype.

Plusieurs expériences ont été réalisées avec différents mouvements de l'utilisateur, des changements d'échelle, et des occultations partielles. La figure 6.13 présente une augmentation concernant une procédure d'assemblage d'un élément de l'objet. Cette animation est activée en réponse à une requête de l'utilisateur, réalisée par un laser. L'ensemble du système fonctionne à 25 images par seconde. Le module de localisation montre de très bonnes performances sous divers scénarios. En comparaison avec l'interaction basée sur le doigt, l'interaction par le laser est moins naturelle mais elle est plus précise, tout particulièrement pour la sélection des composants de petite taille.

6.4 Conclusion

Nous avons montré dans ce chapitre que le cadre de localisation proposé dans cette thèse est parfaitement adapté pour la réalité augmentée. Il permet un vrai développement de la RA sur objet 3D dans différents domaines tels que le *marketing*, l'aide à la vente, la maintenance industrielle, *etc*. Notre solution pour la localisation a subi plusieurs séries de tests sous des conditions réelles et elle s'est avérée très fiable pour différentes applications, notamment pour la personnalisation de cuisine et également dans le domaine automobile pour lequel plusieurs scénarios ont été proposés : l'aide à l'entretien et à la maintenance du moteur, la personnalisation virtuelle de la carrosserie et du tableau de bord de la voiture. Des performances temps réel sont atteintes sur une tablette grand public permettant d'envisager un

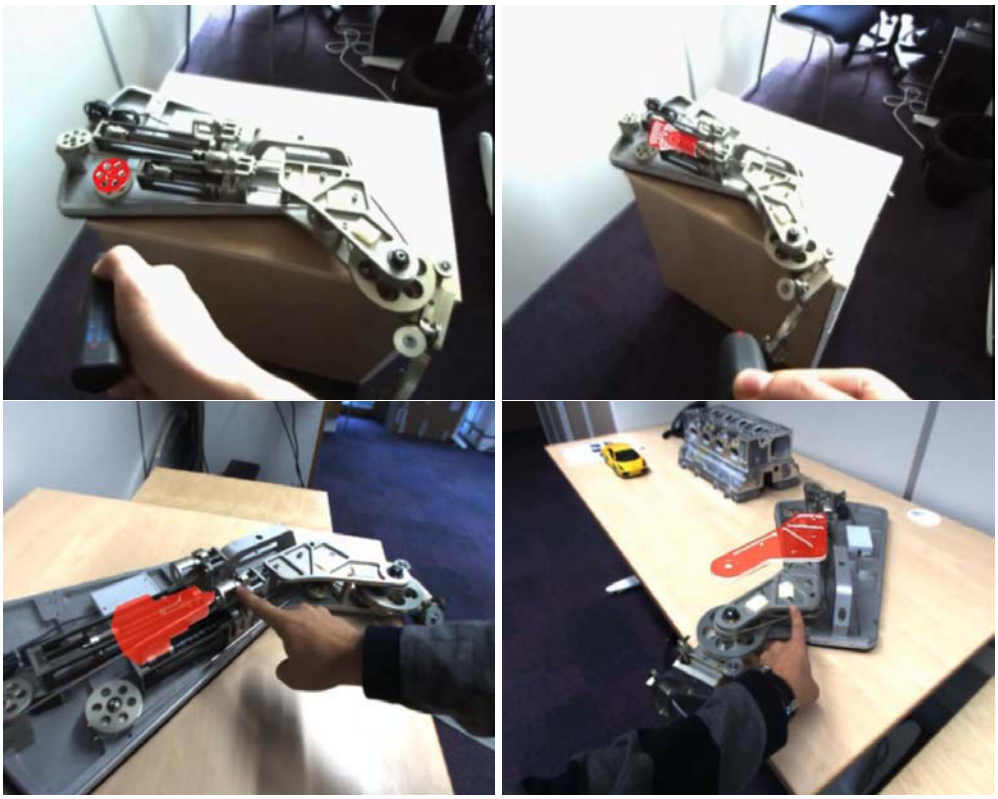


FIGURE 6.13 – Des images de RA avec une interaction de l'utilisateur utilisant un pointeur laser (sur la première ligne) ou un doigt (sur la deuxième ligne).

large déploiement de ces applications de réalité augmentée.

Dans ce chapitre, nous avons également décrit un prototype de RA interactif pour des applications de formations industrielles. Le système fournit sur un casque semi-transparent HMD, des instructions dynamiques de surimpressions pré-enregistrées, en réponse aux interactions de l'utilisateur. Nous avons appliqué notre prototype à une simple démonstration d'une procédure de montage et de démontage virtuel concernant les composants du membre supérieur d'un exosquelette. Nos évaluations expérimentales montrent la robustesse du système aux changements de point de vue ainsi que sa capacité à réaliser, avec une grande précision, un recalage sur un objet industriel peu texturé. En outre, la stabilité et la précision de la localisation de cette solution, permettent d'assurer une interaction rapide, précise et utile pour les applications de formation industrielles.

Conclusion

Travaux réalisés

L'objectif de cette thèse était de proposer une amélioration des solutions existantes pour la localisation d'une caméra monoculaire par rapport à un objet d'intérêt de la scène pour lequel un modèle 3D est disponible. Nous avons proposé un cadre d'unification des principales méthodes de localisation, à savoir le SLAM et le suivi basé modèle avec et sans mise à jour. Cette unification est réalisée à travers le concept d'ajustement de faisceaux contraint par le modèle 3D intégré dans un SLAM basé images clés. Deux approches de localisation, dépendantes de la nature de la contrainte utilisée, en résultent.

La première solution consiste à unifier un SLAM avec un suivi basé modèle. Deux fonctions de coût ont été proposées : la première contraint l'ajustement de faisceaux par des segments 3D extraits d'un modèle géométrique et la seconde le contraint par un nuage de points 3D. Ces fonctions prennent en compte les primitives issues du modèle et les relations multi-vues des primitives du modèle et de l'environnement. Des résultats expérimentaux, sur des données de synthèse, montrent que l'approche proposée fournit de meilleurs résultats, en terme de précision de la localisation que le SLAM « classique ». De même, en terme de stabilité et de robustesse, la comparaison effectuée par la suite, sur des données réelles, entre l'approche proposée et l'état de l'art en suivi d'objet 3D, montre que cette dernière fournit de meilleurs résultats que les algorithmes de suivi basé modèle. En effet, la prise en compte de l'environnement permet de stabiliser la localisation et de conserver le suivi, même si l'objet d'intérêt est peu ou pas visible.

Dans un deuxième temps, nous avons souhaité traiter le cas où nous disposons d'un modèle géométrique peu précis de l'objet d'intérêt. Ce type de modèle peut mettre à défaut la méthode précédente. Nous proposons dans ce cas une approche qui consiste à unifier un SLAM avec un suivi basé modèle avec mise à jour. Deux processus d'ajustement de faisceaux contraint ont été présentés pour les algorithmes pSLAM et sSLAM. Le premier contraint l'ajustement de faisceaux dans le cas où les primitives reconstruites sont des points (pSLAM) et le second dans le cas où les primitives sont des segments (sSLAM). Deux fonctions de coût ont été proposées. Celles-ci prennent en compte les contraintes d'appartenance des primitives de l'objet aux plans du modèle et les relations multi-vues liées à l'observation de l'en-

semble des primitives reconstruites. Des tests sur des données de synthèse et des données réelles montrent l'apport des primitives de l'environnement dans l'ajustement de faisceaux. Cela permet d'augmenter la précision de la reconstruction, la robustesse et le bassin de convergence. Il a donc été montré que l'approche proposée fournit de meilleurs résultats que les méthodes de suivi basé modèle avec mise à jour, qui ne prennent pas en compte les primitives de l'environnement.

Les validations expérimentales dans différents contextes (en intérieur ou extérieur et sur de faibles ou de grands volumes) ont permis de conclure que le cadre de localisation proposé dans cette thèse est parfaitement adapté pour la réalité augmentée. Il permet un développement de la RA sur objet 3D dans différents domaines tels que le marketing, l'aide à la vente, la maintenance industrielle, *etc*.

Perspectives

Les études et expériences réalisées au cours de cette thèse ont permis de mettre en évidence certaines perspectives de nos travaux :

Ajustement de faisceaux avec contraintes multiples. Certains objets font échouer les différentes méthodes de localisation présentées dans cette thèse. Par exemple sur la maquette miniature d'un avion de type A320 illustrée dans la figure 6.14, l'approche SLAM avec l'ajustement de faisceaux LBA_LC&E [6] échoue car il n'y a pas suffisamment de segments 3D qui peuvent être extraits sur l'ensemble du modèle. En effet, le problème avec l'étape d'extraction des segments 3D, est que les parties courbes du modèle d'intérêt ne retournent aucun segment. Sur la maquette d'avion, les segments 3D sont extraits uniquement à partir des ailes latérales et des stabilisateurs horizontaux et verticaux. L'approche SLAM avec l'ajustement de faisceaux pBA_M&E [7] échoue également sur cet exemple car il n'y a pas assez de texture sur la maquette d'avion. La combinaison des deux contraintes permettrait en théorie de résoudre ces problèmes. Le cadre de SLAM contraint proposé dans cette thèse est très flexible et permet de combiner différents types de contraintes entre elles.

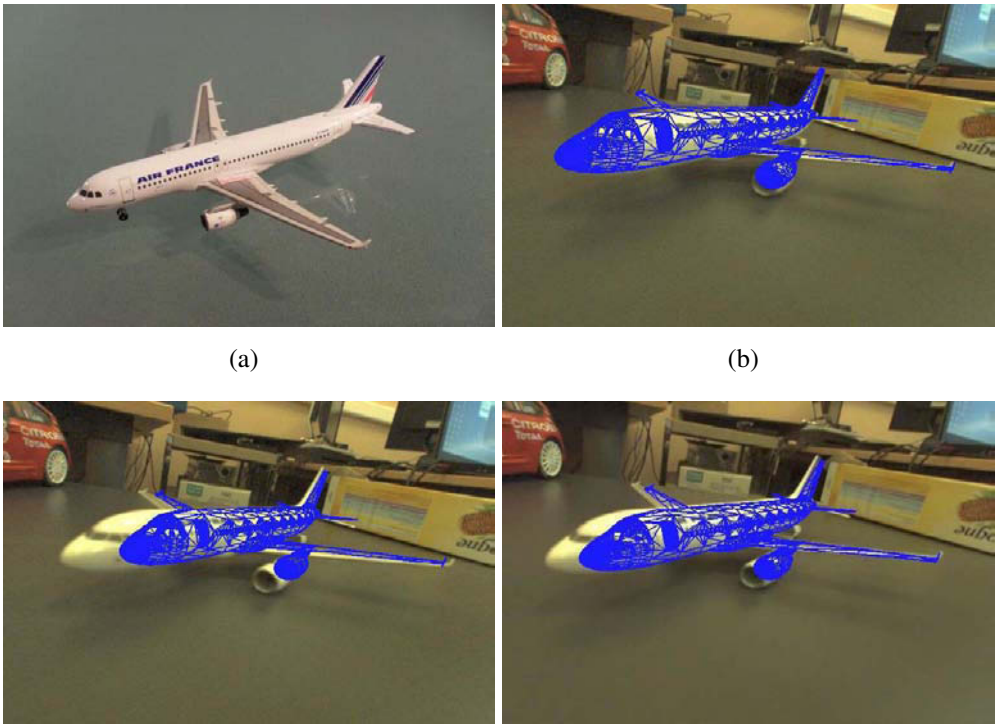
Nous avons alors réalisé une première implémentation de cette solution qui consiste à combiner la contrainte par segments avec la contrainte aux plans. Il en résulte une fonction de coût à trois termes. Ces derniers sont combinés avec une stratégie similaire à celle décrite en section 3.2.5 du chapitre 3. Nous constatons que cet ajustement de faisceaux avec contraintes multiples, fournit de meilleurs résultats que les ajustements de faisceaux contraints par segments ou par les plans du modèle. En effet, comme illustré dans la figure 6.14, ces deux derniers ajustements

6. LBA_LC&E correspond à l'ajustement de faisceaux contraint par des segments 3D extraits du modèle (voir le chapitre 4).

7. pBA_M&E correspond à l'ajustement de faisceaux fondé sur des points contraints aux plans du modèle 3D (voir le chapitre 5).

de faisceaux ne fournissent pas suffisamment de contraintes pour éviter les accumulations d'erreurs du SLAM : le SLAM avec l'algorithme LBA_LC&E (figure 6.14 (c)) et le SLAM avec l'algorithme pBA_M&E (figure 6.14 (d)) ne parviennent pas à une localisation précise durant toute la séquence vidéo.

Les premiers résultats obtenus en combinant les contraintes sont encourageants car cela permet de traiter une plus grande variété d'objets. Cependant une étude plus poussée est nécessaire notamment sur la pondération des contraintes multiples.



(a) (b)

(c) (d)

FIGURE 6.14 – Localisation en environnement partiellement connu composé d'un objet 3D avec des parties courbes. (a) : l'image originale de l'avion miniature. (b) : le résultat de recalage obtenu par le SLAM avec l'ajustement de faisceaux contraint par des segments 3D et les plans du modèle. (c) : le résultat de recalage obtenu par le SLAM avec l'ajustement de faisceaux contraint par des segments et (d) le résultat obtenu par le SLAM avec l'ajustement de faisceaux contraint aux plans du modèle. Notez que le modèle 3D géométrique ne correspond pas exactement à l'avion miniature utilisé dans cette expérimentation : les deux réacteurs ne sont pas à la bonne échelle et les stabilisateurs horizontaux sont mal orientés.

Robustesse à des imprécisions de modèle. Nous avons montré sur quelques exemples que la solution présentée dans ce mémoire est robuste à des imprécisions de modèle. Cette solution permet alors de gérer des modèles reconstruits par

exemple avec le capteur Kinect. Cependant, un travail intéressant serait d'évaluer quantitativement la robustesse à l'imprécision du modèle.

Initialisation du SLAM contraint. Nos futurs travaux se concentreront également sur l'initialisation du système de suivi pour les pièces peu texturées. Prenons par exemple, le cas de la carrosserie de la voiture décrit dans le chapitre 6. A l'heure actuelle, l'étape d'initialisation est gérée par une reconnaissance d'images acquises préalablement lors d'une phase d'apprentissage. Cette reconnaissance porte essentiellement sur la texture de l'environnement de la voiture. La contrepartie est que l'environnement doit être réappris à chaque déplacement de la voiture. Nous étudions une procédure d'initialisation des objets peu texturés basée sur l'extraction, la description et l'appariement de segments, ce qui permettrait de retrouver un point de vue par mise en correspondance de segments. Cela permettrait également de réaliser un sSLAM pour évaluer la méthode d'ajustement de faisceaux de segments contraint aux plans du modèle sur des données réelles.

Ajustement de faisceaux contraint pour des objets courbes. Nous avons vu sur la séquence de l'avion (figure 6.14) que la combinaison des contraintes segments et plans permet de traiter des objets courbes et peu texturés. Une autre solution peut être un ajustement de faisceaux contraint par les contours d'occultations. Les approches (Petit *et al.* (2012); Wuest *et al.* (2007b)) traitent ce problème mais elles manquent de précision et de stabilité. Ces méthodes de localisation basées modèle reposent sur l'utilisation des unités de traitement graphiques (GPU) et des techniques de rendu 3D. Ceci permet de gérer automatiquement la projection du modèle et de déterminer les contours visibles de la scène rendue à partir du point de vue courant. Un avantage de ces techniques est d'extraire dynamiquement (en fonction du rendu) les contours d'occultation pour mieux traiter l'étape de mise en correspondance 3D-2D entre les arêtes du modèle géométrique et les contours de l'image.

Meilleure différenciation des primitives 3D reconstruites. Pour le cas de l'ajustement de faisceaux fondé sur des points contraints aux plans du modèle, l'étape de mise en correspondance entre les points 3D reconstruits en ligne par le SLAM et les plans du modèle est importante. Pour cela, la solution actuelle utilise la rétroprojection des points 2D sur le modèle 3D surfacique. Une autre solution possible est de détecter dans les images les éléments les plus couramment observés sur ce type d'objet. En particulier, Brostow *et al.* (2008) ont montré qu'il est envisageable de réaliser une segmentation de l'image par des techniques d'apprentissage. Dès lors, un point 3D pourrait être associé au modèle ou à l'environnement par la zone dans laquelle se situent ses observations dans les images.

Par ailleurs, la solution actuelle repose sur l'hypothèse que l'objet d'intérêt est rigide et statique dans la scène ce qui est limitant pour beaucoup d'applications comme par exemple la maintenance industrielle pour laquelle cette hypothèse n'est pas garantie de par les démontages et remontages successifs des différentes pièces. Il serait intéressant de généraliser l'approche de localisation par SLAM contraint au modèle à des cas plus complexes avec un objet mobile, articulé et/ou démontable.

Objet articulé. Les approches existantes (Comport *et al.* (2004)) de suivi d'objets articulés considèrent que la caméra est fixe et que la « déformation » de l'objet est lente. Par ailleurs, étendre le SLAM contraint aux objets présentant des articulations permettrait de passer outre ces hypothèses simplificatrices. Pour le cas d'objets articulés, une fonction de coût intégrant la cinématique de l'objet mériterait d'être testée. Notons cependant que l'identification dans l'image des parties mobiles ainsi que l'association des primitives détectées à chaque partie articulée, peut être un point difficile. Combiner les contraintes issues de l'environnement, de l'objet et du modèle CAO articulé permettront alors d'assurer une précision et une stabilité du recalage et ceci même dans le cas d'une caméra en mouvement.

Objet mobile. Concernant l'hypothèse d'objet statique, nous prévoyons également d'étendre le SLAM contraint au cas d'objet en mouvement. Une idée serait de réaliser un SLAM contraint au modèle uniquement sur l'objet mobile. Utiliser l'environnement (statique) de l'objet dans le processus permettra de prédire au mieux le déplacement de la caméra ce qui facilitera ensuite l'estimation du déplacement de l'objet d'intérêt à partir de son modèle 3D géométrique. L'objet d'intérêt et l'environnement continueront à être reconstruits au cours du temps afin de conserver la stabilité du recalage.

ANNEXE

A

Comparaison de différents ajustements de faisceaux fondés sur des points contraints aux plans

Dans cette annexe nous détaillons une fonction de coût pour le cadre d'ajustement de faisceaux contraint aux plans. Celle-ci est basée sur la contrainte homographique. Nous verrons dans ce chapitre qu'elle présente notamment l'avantage de posséder un large bassin de convergence ce qui en fait une solution adaptée aux applications nécessitant de résoudre, sans la contrainte d'un traitement temps réel, un problème d'ajustement de faisceaux global. Cette nouvelle fonction de coût est comparée à celle proposée en section 5.2. Les résultats sur des données réelles et des données de synthèse démontrent l'apport de cette formulation de la contrainte planaire dans le cadre de l'approche proposée. Une partie des travaux décrits dans ce chapitre ont donné lieu à une publication (Tamaazousti et al. (2011b)).

A.1 Ajustement de faisceaux contraint aux plans par homographie

La technique standard pour optimiser la consistance globale d'une reconstruction de type SLAM ou SfM est l'ajustement de faisceaux (BA) qui optimise simultanément les poses des caméras et la structure de la scène. Une alternative moins utilisée est d'optimiser uniquement les poses des caméras à travers les contraintes de la géométrie épipolaire (EG) et ensuite de reconstruire un nuage de points 3D par triangulation. Après quelques rappels sur la géométrie épipolaire, nous allons définir une fonction de coût basée sur cette dernière et pouvant être utilisée pour

optimiser la reconstruction d'un environnement inconnu. Ensuite nous proposerons une nouvelle fonction de coût composée reprenant le principe de l'ajustement de faisceaux contraint fondé sur des points contraints aux plans.

A.1.1 Optimisation utilisant la géométrie épipolaire (EG)

Rappelons que la géométrie épipolaire définit la relation entre les images acquises par deux poses d'une caméra observant la même scène à partir de points de vue différents. La géométrie épipolaire (voir Hartley et Zisserman (2000) pour plus de détails) lie deux observations $(\mathbf{q}_{i,1}, \mathbf{q}_{i,2})$ d'un point 3D $\mathbf{Q}_i$ à travers la matrice Fondamentale : $\mathbf{q}_{i,2}^{\mathsf{T}} \mathsf{F} \mathbf{q}_{i,1} = 0$, où F est une matrice 3×3 de rang 2. Cette relation signifie qu'un point $\mathbf{q}_{i,2}$ dans la seconde image, apparié au point $\mathbf{q}_{i,1}$ dans la première, se trouve sur la ligne épipolaire $\mathbf{l} = \mathsf{F}\mathbf{q}_{i,1}$. La matrice Fondamentale dépend directement du déplacement relatif entre deux poses de la caméra. On écrit alors la relation suivante : $\mathsf{F} = \mathsf{K}^{-\mathsf{T}} \left[\mathbf{t}_{1\to2}\right]_\times \mathsf{R}_{1\to2} \mathsf{K}^{-1}$. Ce déplacement inter-caméras peut alors être optimisé (voir Simon *et al.* (1998)) en minimisant le critère non linéaire suivant : $\mathcal{E}\left((\mathsf{R}, \mathbf{t})_{1\to2}\right) = \sum_{i=1}^{N} d_l^2(\mathbf{q}_{i,2}, \mathsf{F}_{2,1}\mathbf{q}_{i,1}) + d_l^2(\mathbf{q}_{i,1}, \mathsf{F}_{1,2}\mathbf{q}_{i,2})$, où $d_l(\mathbf{q}, \mathbf{l})$ est la distance point-droite entre le point $\mathbf{q}$ et la droite $\mathbf{l}$, telle que $d_l^2(\mathbf{q}, \mathbf{l}) = \frac{\left(\mathbf{q}^{\mathsf{T}}\mathbf{l}\right)^2}{\|\mathbf{l}\|^2 w^2}$. Ce principe peut être étendu au cas multi-vues. L'estimation du déplacement de la caméra est alors optimisée en minimisant la fonction de coût suivante :

$$\mathcal{E}\left(\left\{(\mathsf{R}, \mathbf{t})_{p\to p+1}\right\}_{p=1}^{m-1}\right) = \sum_{i=1}^{N} \sum_{j\in A_i} \sum_{k\in\mathcal{A}_i}^{k\neq j} d_l^2(\mathbf{q}_{i,j}, \mathsf{F}_{j,k}\mathbf{q}_{i,k}), \tag{A.1}$$

où $\mathsf{F}_{j,k}$ est la matrice Fondamentale entre la paire d'images (j, k).

Notons que dans le cas d'une scène plane l'homographie est l'équivalent de la matrice Fondamentale. Deux images observant un même plan π sont liées par une homographie H. Soit $\mathbf{q}_{i,1}$ l'observation d'un point $\mathbf{Q}_i \in \pi$ dans la première vue et $\mathbf{q}_{i,2}$ l'observation dans la seconde vue, alors $\mathbf{q}_{i,1} \sim \mathsf{H}\mathbf{q}_{i,2}$.

L'homographie H induite par le plan π est donnée par :

$$\mathsf{H} = \mathsf{K}(\mathsf{R} - \frac{\mathbf{t}\mathbf{n}^{\mathsf{T}}}{d})\mathsf{K}^{-1}, \tag{A.2}$$

où, $\mathbf{n}$ représente la normale du plan et d la distance entre C_1 et le plan. Cette relation peut être utilisée pour optimiser une reconstruction initiale de type SLAM. La fonction de coût suivante est minimisée :

$$\mathcal{E}\left(\left\{(\mathsf{R}, \mathbf{t})_{p\to p+1}\right\}_{p=1}^{m-1}\right) = \sum_{i=1}^{N} \sum_{j\in\mathcal{A}_i} \sum_{k\in\mathcal{A}_i}^{k\neq j} d^2(\mathbf{q}_{i,j}, \mathsf{H}_{j,k}^{\pi_i}\mathbf{q}_{i,k}), \tag{A.3}$$

où $\mathsf{H}_{j,k}^{\pi_i}$ est l'homographie induite par l'observation du plan π_i par les caméras j et k. Notons que cette fonction de coût ne prend pas en compte les points 3D. Seules les poses de caméras sont optimisées.

A.1.2 Choix dans les combinaisons des termes

Nous avons présenté dans le chapitre 5 différentes fonctions de coût pouvant être utilisées, soit pour les points associés à l'environnement soit pour ceux associés aux plans du modèle. Il y a donc plusieurs combinaisons possibles entre ces fonctions pour réaliser un ajustement de faisceaux contraint aux plans. Par souci de cohérence du choix des combinaisons nous ne présentons ici que deux fonctions de coût composées. La première, proposée dans ce chapitre, utilise les contraintes homographiques (l'équation (A.3)) pour la partie de l'environnement associé au modèle et leurs équivalents dans le cas d'une structure inconnue qui sont les contraintes liées à la géométrie épipolaire (l'équation (A.1)). Elles ont en commun le fait de ne minimiser que les déplacements relatifs entre deux caméras. La fonction de coût composée, ainsi formulée, est décrite par l'équation (A.4). La seconde fonction de coût est composée des équations (1.51) et (5.3) qui utilisent explicitement les points 3D dans un ajustement de faisceaux. Ainsi, les points 3D associés au modèle 3D ont uniquement deux degrés de liberté alors que les points 3D de la partie inconnue de l'environnement ont trois degrés de liberté. Nous rappelons dans l'équation (A.5) cette dernière fonction de coût mais elle est décrite en détail en section 5.2 du chapitre 5.

$$\begin{aligned}\mathcal{E}\left(\left\{(\mathsf{R},\mathbf{t})_{p\to p+1}\right\}_{p=1}^{N_c-1}\right) &= \underbrace{\sum_{i\in\mathcal{U}}\sum_{j\in A_i}\sum_{k\in A_i}^{k\neq j}\rho\left(d_l^2(\mathbf{q}_{i,j},\mathsf{F}_{j,k}\mathbf{q}_{i,k}),c_1\right)}_{\text{Partie inconnue de l'environnement }(\mathcal{E}_E)} \\ &+ \underbrace{\sum_{i\in\mathcal{M}}\sum_{j\in\mathcal{A}_i}\sum_{k\in\mathcal{A}_i}^{k\neq j}\rho\left(d^2(\mathbf{q}_{i,j},\mathsf{H}_{j,k}^{\pi_i}\mathbf{q}_{i,k}),c_2\right)}_{\text{Partie connue de l'environnement }(\mathcal{E}_M)}\end{aligned} \tag{A.4}$$

$$\begin{aligned}\mathcal{E}\left(\{\mathsf{R}_j,\mathbf{t}_j\}_{j=1}^{N_c},\{\mathbf{Q}_i\}_{i\in\mathcal{U}},\{\mathbf{Q}_i^{\pi_i}\}_{i\in\mathcal{M}}\right) &= \underbrace{\sum_{i\in\mathcal{U}}\sum_{j\in\mathcal{A}_i}\rho\left(d^2(\mathbf{q}_{i,j},\mathsf{P}_j\mathbf{Q}_i),c_1\right)}_{\text{Partie inconnue de l'environnement }(\mathcal{E}_E)} \\ &+ \underbrace{\sum_{i\in\mathcal{M}}\sum_{j\in\mathcal{A}_i}\rho\left(d^2(\mathbf{q}_{i,j},\mathsf{P}_j\mathsf{M}^{\pi_i}\mathbf{Q}_i^{\pi_i}),c_2\right)}_{\text{Partie connue de l'environnement }(\mathcal{E}_M)}\end{aligned} \tag{A.5}$$

A.2 Evaluation sur des données de synthèse

Dans cette section, nous comparons, sur la séquence de synthèse « cube », quatre algorithmes d'ajustement de faisceaux contraint pour le pSLAM : pBA_M, pBA_M&E, pEG_M et pEG_M&E [1]. Ils minimisent respectivement les équations

1. p signifie qu'il s'agit de primitives de type points (pSLAM), M signifie que seules les contraintes au modèle (c'est-à-dire de la partie connue de l'environnement) sont utilisées tandis que

(5.3), (A.5), (A.3) et (A.4) en suivant la procédure décrite dans le tableau 5.3 de la section 5.5. Notons que pour pBA_M et pEG_M le M-estimateur de Geman-Mclure (voir la section 3.2.4) est aussi utilisé pour gérer les mauvaises associations entre les points et les plans du modèle.

Nous réalisons ici une comparaison des quatre algorithmes décrits précédemment dans le cas d'un ajustement de faisceaux global. La comparaison est réalisée en terme de bassin de convergence, de vitesse de convergence, sur la précision de la reconstruction obtenue ainsi que sur la robustesse à un mauvais recalage initial.

Comparaison des quatre algorithmes d'optimisation non linéaire. Nous comparons ici les quatre algorithmes (pEG_M, pBA_M, pEG_M&E et pBA_M&E) en simulant différentes sources d'erreurs comme celle du recalage initial, de la dérive, *etc*. A partir des poses de caméras de la vérité terrain, un nuage de points 3D éparse est généré par triangulation de points d'intérêt appariés au cours de la séquence. Deux types de perturbations sont alors générés sur cette reconstruction initiale.

- ▷ Le premier test (TEST RIGIDE) simule des imprécisions du recalage initial entre les repères du monde et du modèle. Pour cela, une perturbation rigide est appliquée sur la reconstruction globale (caméras et points 3D).
- ▷ Le deuxième test (TEST NON RIGIDE) simule des erreurs d'estimation du déplacement relatif entre deux caméras. Ces erreurs proviennent de l'estimation, par le SLAM, du déplacement de la caméra en présence de bruit, de valeurs aberrantes, d'erreurs de calcul numérique, *etc*. Comme ces erreurs ne sont pas constantes dans le temps, une perturbation non rigide de la reconstruction globale est réalisée en perturbant de manière aléatoire les déplacements inter-caméras et en régénérant par la suite un nuage de points 3D par triangulation.

Les 50 tirages aléatoires sont réalisés pour chaque amplitude de perturbation. Cette dernière varie de 1% à 10% du rayon du cercle décrivant la trajectoire de la caméra. Nous appliquons alors les quatre algorithmes sur les reconstructions ainsi obtenues. La qualité de la reconstruction finale est mesurée par le RMS 3D sur les positions de la caméra entre celles de la vérité terrain et celles estimées. Les quatre algorithmes d'ajustement de faisceaux sont comparés en termes de précision, de vitesse et de fréquence de convergence. La précision est donnée par la valeur du RMS 3D uniquement lorsqu'il y a convergence de l'algorithme. La fréquence de convergence est le pourcentage de tirages pour lesquels le RMS 3D a diminué. Quant à la vitesse de convergence, elle est mesurée par l'évolution de l'erreur 3D au cours des itérations successives du Levenberg-Marquardt, pour une perturbation donnée (4% dans nos expériences). Les résultats présentés sont des moyennes sur les 50 tirages aléatoires.

M&E signifie que les contraintes au modèle ainsi que les informations du reste de l'environnement sont, à la fois, prises en compte.

Fréquence de convergence. Entre les deux tests, TEST RIGIDE et TEST NON RIGIDE les quatre algorithmes ont un comportement similaire. Les algorithmes pEG_M&E et pBA_M&E ont le bassin de convergence le plus large alors que pBA_M a le plus petit bassin de convergence. Pour un déplacement d'amplitude 10% lors du TEST NON RIGID, pEG_M&E et pBA_M&E convergent approximativement dans tous les cas, alors que pEG_M converge a 60% et pBA_M ne converge jamais.

Précision. pEG_M&E et pBA_M&E sont les algorithmes les plus précis pour des perturbations rigides et non rigides. Ils sont suivis de près par pEG_M alors que pBA_M a les plus mauvaises performances. Par exemple, pour une amplitude de perturbation de 8% pour le TEST RIGIDE, les RMS 3D des algorithmes pEG_M&E et pBA_M&E sont inférieurs à 5cm. Le RMS 3D est supérieur à 5cm pour pEG_M et il est autour de 11cm pour pBA_M.

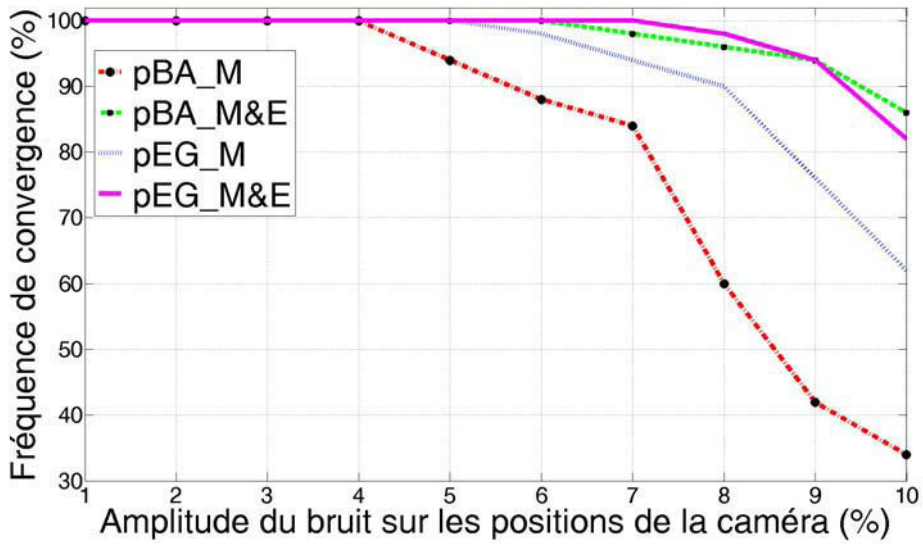
Vitesse de convergence. pBA_M&E converge plus vite que les autres algorithmes. Pour TEST RIGIDE, après trois itérations pBA_M&E réduit l'erreur 3D par un facteur de 2.7, pEG_M&E par un facteur de 2 et seulement par un facteur de 1.5 pour pEG_M et pBA_M.

Résumé. Les algorithmes pBA_M&E et pEG_M&E sont plus performants que pBA_M et pEG_M en termes de précision, de vitesse et de fréquence de convergence. Cela montre que la prise en compte des points 3D de la partie inconnue de l'environnement dans l'optimisation non linéaire, améliore de manière remarquable les résultats de localisation. De plus, pEG_M&E et pEG_M sont respectivement plus performants que pBA_M&E et pBA_M, notamment en terme de bassin de convergence. Cela montre que l'optimisation avec les contraintes liées à la géométrie épipolaire est plus adaptée pour un problème d'ajustement de faisceaux global.

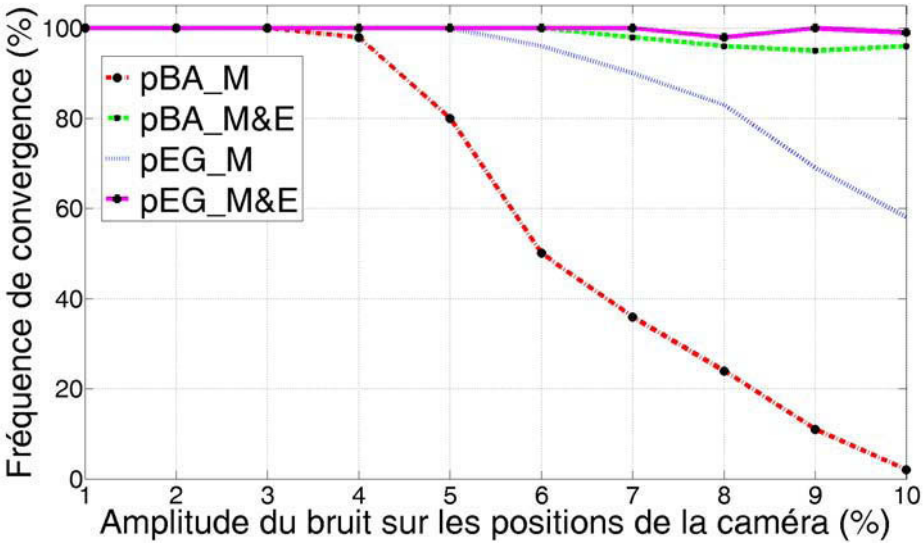
A.3 Evaluation sur des données réelles

Dans cette section nous évaluons les algorithmes d'ajustement de faisceaux contraint utilisant la géométrie épipolaire sur des données réelles pour une application à la localisation dans un grand environnement. Il s'agit de réaliser un ajustement de faisceaux global pour une application de localisation en milieu urbain. Pour cela, une séquence réelle a été réalisée avec une caméra IEEE1394 GUPPY fournissant des images (640×480) à une fréquence de 30 images par seconde.

Des travaux récents, par exemple Lothe *et al.* (2009), ont montré que dans un contexte urbain on peut utiliser un modèle de type SIG pour contraindre le nuage de points provenant d'un algorithme de type SLAM afin d'obtenir une reconstruction plus précise. Dans ce papier, ils proposent un processus hors ligne en deux étapes

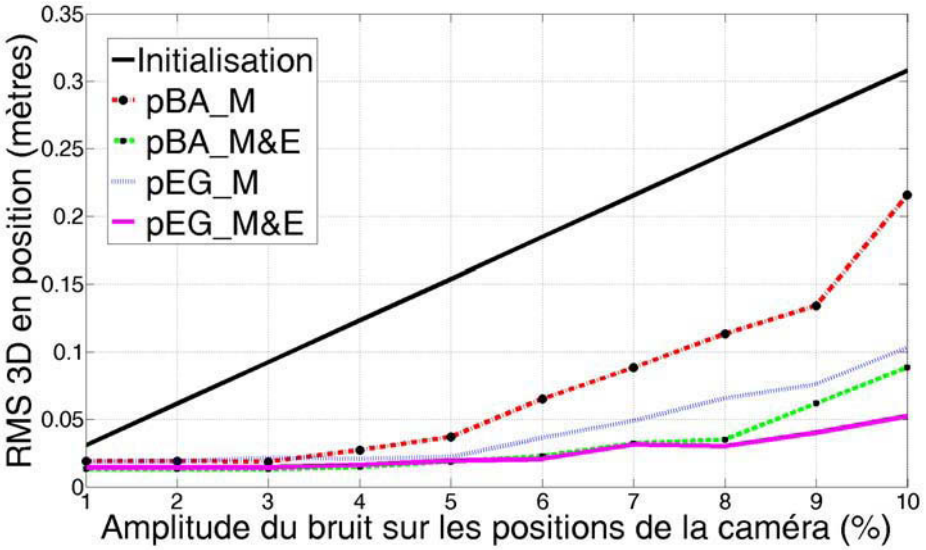


(a)

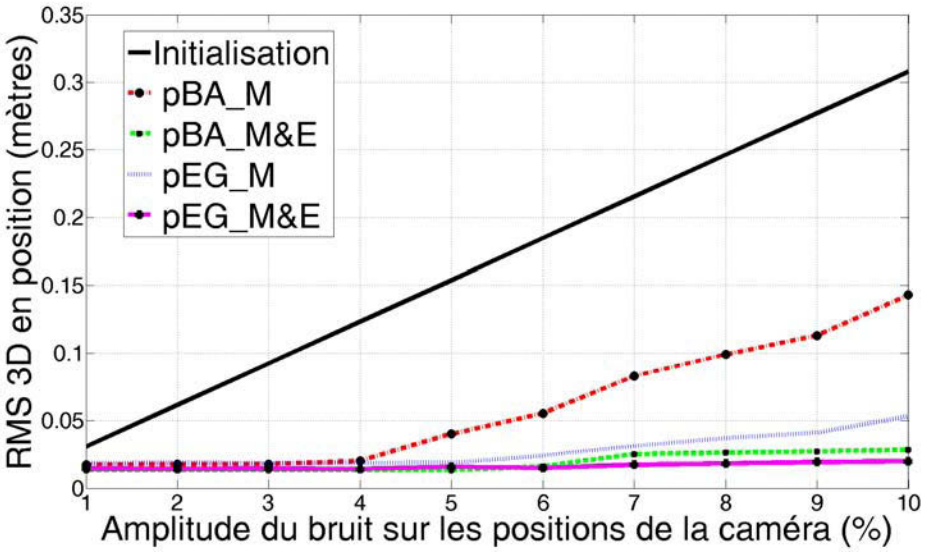


(b)

FIGURE A.1 – **Evaluation de la fréquence de convergence des quatre algorithmes (pEG_M, pBA_M, pEG_M&E et pBA_M&E).** Résultats obtenus avec les quatre algorithmes d'ajustement de faisceaux contraint aux plans pour le TEST RIGIDE (a) et le TEST NON RIGIDE (b).

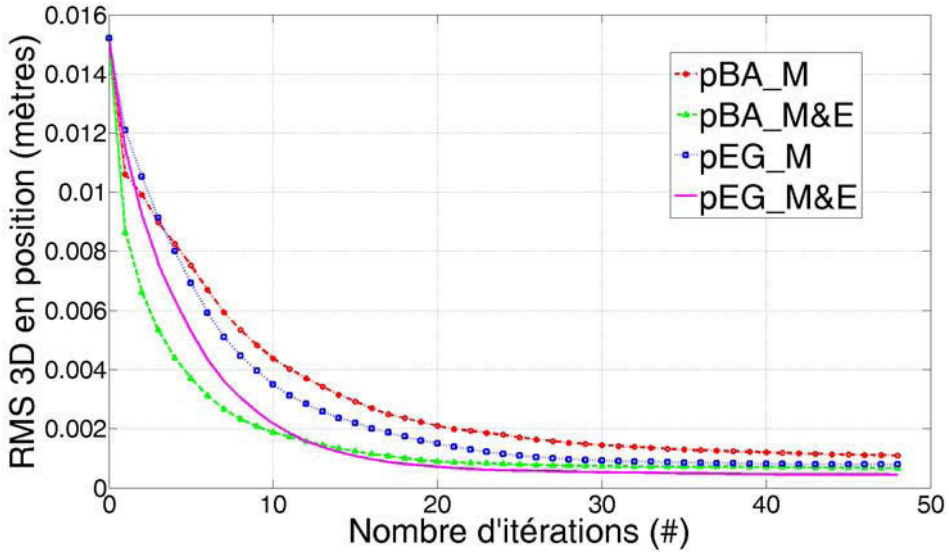


(a)

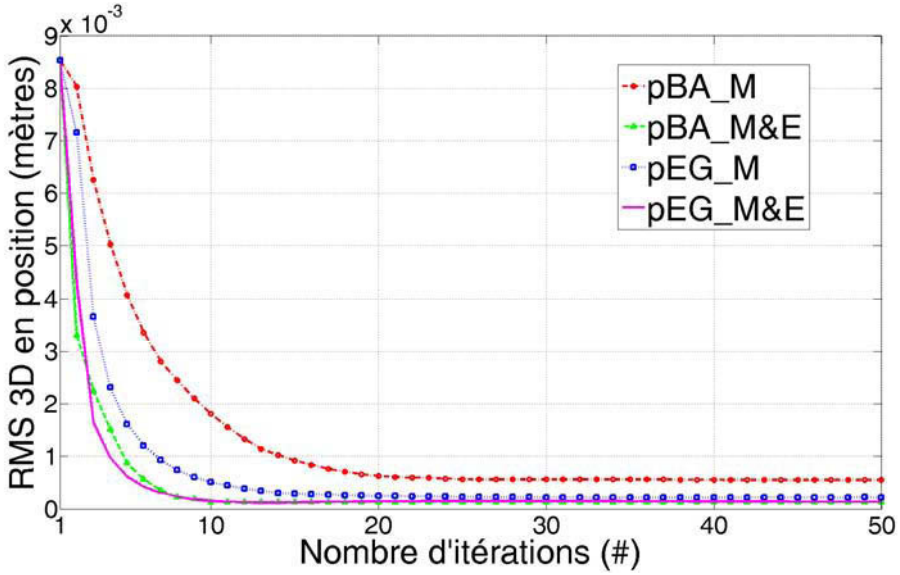


(b)

FIGURE A.2 – **Evaluation de la précision des quatre algorithmes (pEG_M, pBA_M, pEG_M&E et pBA_M&E).** Résultats obtenus avec les quatre algorithmes d'ajustement de faisceaux contraint aux plans pour le TEST RIGIDE (a) et le TEST NON RIGIDE (b).



(a)



(b)

FIGURE A.3 – **Evaluation de la vitesse de convergence des quatre algorithmes (pEG_M, pBA_M, pEG_M&E et pBA_M&E).** Résultats obtenus avec les quatre algorithmes d'ajustement de faisceaux contraint aux plans pour le TEST RIGIDE (a) et le TEST NON RIGIDE (b).

qui aligne dans un premier temps, approximativement, la reconstruction sur le modèle avec un ICP non rigide et qui par la suite, améliore la reconstruction avec une optimisation non linéaire basée modèle. Notons que la partie inconnue de l'environnement, c'est-à-dire tout ce qui n'est pas un bâtiment, n'est pas prise en compte dans leur processus d'optimisation. Leur algorithme d'optimisation non linéaire est similaire à pEG_M avec une fonction de coût légèrement différente.

Ce type d'algorithme d'optimisation n'est évidemment pas robuste lorsque la partie connue de l'environnement est occultée ou absente. Cependant, ce cas de figure est très commun en milieu urbain : il n'y a pas des bâtiments de chaque côté de la chaussée dans toutes les rues et ils peuvent être occultés par des bus, des camions *etc*. Nous montrerons par la suite que le fait d'intégrer la partie inconnue de l'environnement (la route, les arbres, *etc.*), dans le processus de minimisation, permet de résoudre ces problèmes.

Une séquence vidéo de 975 images a été acquise au cours d'un déplacement en voiture de 500 mètres. La reconstruction initiale de type SLAM a été approximativement alignée sur le modèle avec une correction par ICP non rigide comme dans Lothe *et al.* (2009). Elle est alors raffinée par les algorithmes pEG_M&E et pEG_M. Les reconstructions obtenues sont alors évaluées, qualitativement, par une relocalisation en ligne.

Optimisation non linéaire hors ligne. La figure A.4 illustre le nuage de points 3D et la trajectoire de la caméra obtenus après le raffinement par les algorithmes pEG_M et pEG_M&E. Les principales différences entre les deux sont localisées à deux endroits de la scène et ont été entourées et zoomées. Dans le premier cas, la caméra au début du carrefour n'observe pas de bâtiment et pour le deuxième cas, les bâtiments sur le côté gauche sont occultés par un bus. Dans ces deux cas critiques avec l'algorithme pEG_M, la caméra présente une trajectoire improbable et la structure du nuage de points est aussi erronée (en rouge sur la figure A.4). L'algorithme pEG_M&E fournit une trajectoire régulière, en particulier dans la première zone où la structure du mur semble être rétablie (en bleu sur la figure A.4). Ces résultats montrent qu'en utilisant la partie inconnue de l'environnement, la reconstruction est, dans l'ensemble, de meilleur qualité. Ces deux reconstructions sont alors utilisées pour former deux bases de données qui encodent les descripteurs de l'ensemble des points 3D [2] sous la forme d'un arbre de vocabulaire (Nistér et Stewenius (2006)).

Relocalisation en ligne. Une autre séquence vidéo de 649 images a été réalisée en suivant approximativement la même trajectoire. L'environnement a légèrement changé depuis la précédente séquence (par exemple des voitures garées sur la chaussée). Pour chaque image de la séquence une relocalisation est effectuée. Pour cela

2. Les points 3D associés à la partie connue et inconnue de l'environnement sont conservés pour les deux cartes.

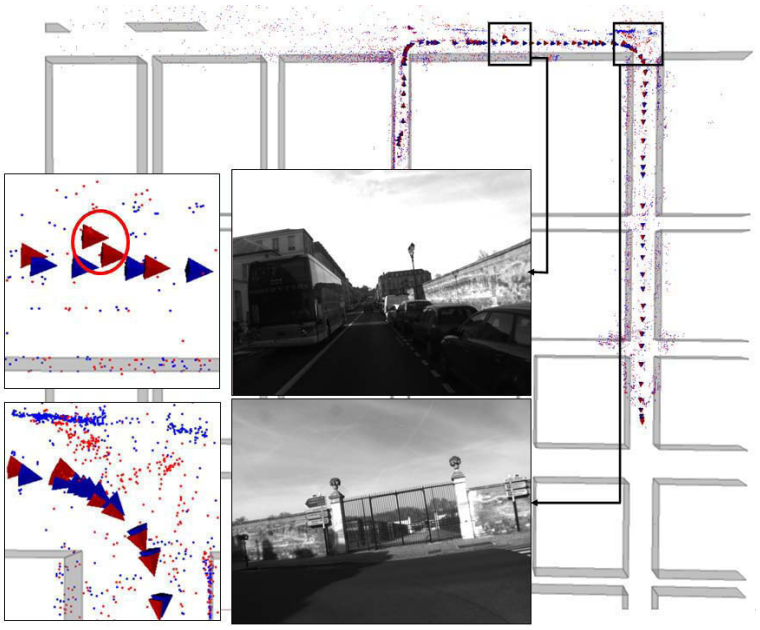


FIGURE A.4 – **Reconstruction par SLAM en milieu urbain.** En bleu (resp. rouge) la reconstruction obtenue après optimisation avec l'algorithme pEG_M&E (resp. pEG_M). Les cercles rouges mettent en avant les poses improbables de la caméra introduites par l'algorithme pEG_M.

une mise en correspondance entre les descripteurs extraits de l'image courante et ceux de la base de donnée est effectuée. La pose de la caméra est alors calculée avec l'estimateur RANSAC sur les appariements. Le nombre de relocalisations retenues est de 539 et 642 pour les bases de données obtenues respectivement avec les algorithmes pEG_M et pEG_M&E. Avec la première base de donnée, des échecs de relocalisation apparaissent principalement dans les deux parties de la séquence décrite précédemment. De plus, le nombre de correspondances 3D-2D utilisées pour calculer la pose de la caméra est, en moyenne sur la séquence, de 50 pour pEG_M contre 60 pour pEG_M&E. Nous pouvons alors conclure que l'ensemble des points 3D est plus cohérent pour la reconstruction raffinée par l'algorithme pEG_M&E. Cela montre que ce dernier fournit une reconstruction de meilleure qualité.

FIGURE A.5 – **Ajustement de faisceaux contraint aux plans pour une application de réalité augmentée pour l'aide à la navigation en milieu urbain.** A gauche : un exemple d'échec de relocalisation avec la reconstruction optimisée par l'algorithme pEG_M. A droite : pour la même image, la relocalisation est possible avec la reconstruction raffinée par l'algorithme pEG_M&E. Cela est mis en avant à travers une application de réalité augmentée pour l'aide à la navigation.

Bibliographie

Arvika. www.arvika.de.

F. ABABSA et M. MALLEM : Robust line tracking using a particle filter for camera pose estimation. *In Virtual Reality Software and Technology*, VRST, 2006.

S. AGARWAL, N. SNAVELY, I. SIMON, S. M SEITZ et R. SZELISKI : Building rome in a day. *In International Conference on Computer Vision*, ICCV, 2009.

A. ANGELI, S. DONCIEUX, J.A. MEYER et D. FILLIAT : Real-time visual loop-closure detection. *In International Conference on Robotics and Automation*, ICRA, 2008.

M. ARMSTRONG et A. ZISSERMAN : Robust object tracking. *In Asian Conference on Computer Vision*, ACCV, 1995.

S.Y. BAO et S. SAVARESE : Semantic structure from motion. *In Computer Vision and Pattern Recognition*, CVPR, 2011.

A. BARTOLI : *Reconstruction et alignement en vision 3D : points, droites, plans et caméras*. Thèse de doctorat, Institut National Polytechnique de Grenoble, 2003.

A. BARTOLI, V. GAY-BELLILE, U. CASTELLANI, J. PEYRAS, S. OLSEN et P. SAYD : Coarse-to-fine low-rank structure-from-motion. *In Computer Vision and Pattern Recognition*, CVPR, 2008.

A. BARTOLI et P. STURM : Constrained structure and motion from multiple uncalibrated views of a piecewise planar scene. *International Journal of Computer Vision*, 52(1):45–64, 2003.

A. BARTOLI et P. STURM : Structure-from-motion using lines : Representation, triangulation, and bundle adjustment. *Computer Vision and Image Understanding*, 100(3):416–441, 2005.

H. BAY, A. ESS, T. TUYTELAARS et L. VAN GOOL : Surf : Speeded up robust features. *In European Conference on Computer Vision*, ECCV, 2006.

S. BENHIMANE et E. MALIS : Integration of euclidean constraints in template based visual tracking of piecewise-planar scenes. *In International Conference on Intelligent Robots and Systems*, IROS, 2006.

B. BESBES, S. NAUDET-COLLETTE, M. TAMAAZOUSTI, S. BOURGEOIS et V. GAY-BELLILE : An interactive augmented reality system : a prototype for industrial maintenance training applications. *In International Symposium on Mixed and Augmented Reality*, ISMAR, 2012.

G. BLESER, Y. PASTARMOV et D. STRICKER : Real-time 3d camera tracking for industrial augmented reality applications. *In International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, WSCG, 2005.

G. BLESER, H. WUEST et D. STRICKER : Online camera pose estimation in partially known and dynamic scenes. *In International Symposium on Mixed and Augmented Reality*, ISMAR, 2006.

S. BOUGNOUX : From projective to euclidean space under any practical situation, a criticism of self-calibration. *In International Conference on Computer Vision*, ICCV, 1998.

P. BOUTHEMY : A maximum likelihood framework for determining moving edges. *Transactions on Pattern Analysis and Machine Intelligence*, 11(5):499–511, 1989.

G. BROSTOW, J. SHOTTON, J. FAUQUEUR et R. CIPOLLA : Segmentation and recognition using structure from motion point clouds. *In European Conference on Computer Vision*, ECCV, 2008.

M. CALONDER, V. LEPETIT, C. STRECHA et P. FUA : Brief : Binary robust independent elementary features. *In European Conference on Computer Vision*, ECCV, 2010.

J CANNY : A computational approach to edge detection. *Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

G. CARON, A. DAME, et E. MARCHAND : L'information mutuelle pour l'estimation visuelle directe de pose. *In Confèrence Reconnaissance des Formes et Intelligence Artificielle*, RFIA, 2012.

J.A. CASTELLANOS, J.M.M. MONTIEL, J. NEIRA et J.D. TARDÓS : The spmap : A probabilistic framework for simultaneous localization and map building. *Transactions on Robotics and Automation*, 15(5):948–953, 1999.

J.A. CASTELLANOS, J. NEIRA et J.D. TARDÓS : Multisensor fusion for simultaneous localization and map building. *Transactions on Robotics and Automation*, 17(6):908–914, 2001.

H. CHUI et A. RANGARAJAN : A new point matching algorithm for non-rigid registration. *Computer Vision and Image Understanding*, 89(2):114–141, 2003.

J. CIVERA, O. GARCIA, A.J. DAVISON et J.M.M. MONTIEL : 1-point ransac for ekf-based structure from motion. *In International Conference on Intelligent Robots and Systems*, IROS, 2009.

A.I. COMPORT, E. MARCHAND et F. CHAUMETTE : A real-time tracker for markerless augmented reality. *In International Symposium on Mixed and Augmented Reality*, ISMAR, 2003.

A.I. COMPORT, E. MARCHAND et F. CHAUMETTE : Complex articulated object tracking. *In Articulated Motion and Deformable Objects*, AMDO, 2004.

A.I. COMPORT, E. MARCHAND, M. PRESSIGOUT et F. CHAUMETTE : Real-time markerless tracking for augmented reality : the virtual visual servoing framework. *Transaction on Visualization and Computer Graphics*, 12(4):615–628, 2006.

P. DAVID, D. DEMENTHON, R. DURAISWAMI et H. SAMET : Softposit : Simultaneous pose and correspondence determination. *International Journal of Computer Vision*, 59(3):259–284, 2004.

A.J. DAVISON, I.D. REID, N.D. MOLTON et O. STASSE : Monoslam : Real-time single camera slam. *Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1052–1067, 2007.

D.F. DEMENTHON et L.S. DAVIS : Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15(1-2):123–141, 1995.

G. DISSANAYAKE, P. NEWMAN, S. CLARK, H.F. DURRANT-WHYTE et M. CSORBA : A solution to the simultaneous localization and map building (slam) problem. *Transactions on Robotics and Automation*, 17(3):229–241, 2001.

G. DISSANAYAKE, S.B. WILLIAMS, H.F DURRANT-WHYTE et T. BAILEY : Map management for efficient simultaneous localization and mapping (slam). *Autonomous Robots*, 12(3):267–286, 2002.

T. DRUMMOND et R. CIPOLLA : Real-time visual tracking of complex structures. *Transactions on Pattern Analysis and Machine Intelligence*, 24(7):932–946, 2002.

E. EADE et T. DRUMMOND : Edge landmarks in monocular slam. *In British Machine Vision Conference*, BMVC, 2006a.

E. EADE et T. DRUMMOND : Scalable monocular slam. *In Conference on Computer Vision and Pattern Recognition*, CVPR, 2006b.

C. ENGELS, H. STEWÈNIUS et D. NISTÈR : Bundle adjustment rules. *In Photogrammetric Computer Vision (PCV)*, ISPRS, 2006.

A. EUDES et M. LHUILLIER : Error propagations for local bundle adjustment. *In Computer Vision and Pattern Recognition*, CVPR, 2009.

A. EUDES, S. NAUDET-COLLETTE, M. LHUILLIER et M. DHOME : Weighted local bundle adjustment and application to odometry and visual slam fusion. *In British Machine Vision Conference*, BMVC, 2010.

M. FARENZENA, A. BARTOLI et Y. MEZOUAR : Efficient camera smoothing in sequential structure-from-motion using approximate cross-validationt. *In European Conference on Computer Vision*, ECCV, 2008.

I. FARKHATDINOV et J-H. RYU : Development of educational system for automotive engineering based on augmented reality. *In International Conference on Engineering Education and Research*, ICEER, 2009.

O. FAUGERAS : What can be seen in three dimensions with an uncalibrated stereo rig ? *In European Conference on Computer Vision*, ECCV, 1992.

O. FAUGERAS : *Three-dimensional computer vision*. MIT press, 1993.

M.A. FISCHLER et R.C. BOLLES : Random sample consensus : A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

J.-M. FRAHM, P. FITE-GEORGEL, D. GALLUP, T. JOHNSON, R. RAGURAM, C. WU, Y.-H. JEN, E. DUNN, B. CLIPP, S. LAZEBNIK et M. POLLEFEYS : Building rome on a cloudless day. *In European Conference on Computer Vision*, ECCV, 2010.

J. GATES, M. HASEYAMA et H. KITAJIMA : A new conic section extraction approach and its applications. *Transactions on Information and Systems*, 88 (2):239–251, 2005.

V. GAY-BELLILE, S. BOURGEOIS, M. TAMAAZOUSTI, S NAUDET-COLLETTE et S. KNODEL : A mobile markerless augmented reality system for the automotive field. *In International workshop on Tracking Methods and Applications, ISMAR Workshop*, TMA, 2012.

V. GAY-BELLILE, P. LOTHE, S. BOURGEOIS, E. ROYER et S. NAUDET-COLETTE : Augmented reality in large environments : Application to aided navigation in urban context. *In International Symposium on Mixed and Augmented Reality*, ISMAR, 2010a.

V. GAY-BELLILE, M. TAMAAZOUSTI, R. DUPONT et S. NAUDET-COLLETTE : A vision-based hybrid system for real-time accurate localization in an indoor environment. *In International Conference on Computer Vision Theory and Applications*, VISAPP, 2010b.

A.P. GEE et W. MAYOL-CUEVAS : Real-time model-based slam using line segments. *In International Symposium on Visual Computing*, ISVC, 2006.

D. B. GENNERY : Visual tracking of known three-dimensional objects. *International Journal of Computer Vision*, 7(3):243–270, 1992.

I. GORDON et D.G. LOWE : What and where : 3d object recognition with accurate pose. *In Toward Category-Level Object Recognition*, pages 67–82, 2006.

F.S. GRASSIA : Practical parameterization of rotations using the exponential map. *Journal of Graphics Tools*, 3(3):29–48, 1998.

J GRÜNERT : Das pothenotische problem in erweiterter gestalt nebst über seine anwendungen in der geodäsie. *Grunerts Archiv für Mathematik und Physik*, 1: 238–248, 1841.

F.R. HAMPEL, E.M. RONCHETTI, P.J. ROUSSEEUW et W.A. STAHEL : *Robust statistics : the approach based on influence functions*. Wiley series in probability and mathematical statistics. 1986.

B.M. HARALICK, C.N. LEE, K. OTTENBERG et M. NÖLLE : Review and analysis of solutions of the three point perspective pose estimation problem. *International Journal of Computer Vision*, 13(3):331–356, 1994.

R.M. HARALICK, H. JOO, C. LEE, X. ZHUANG, V.G. VAIDYA et M.B. KIM : Pose estimation from corresponding point data. *Transactions on Systems, Man and Cybernetics*, 19(6):1426–1446, 1989.

R.M. HARALICK, D. LEE, K. OTTENBURG et M. NOLLE : Analysis and solutions of the three point perspective pose estimation problem. *In Computer Vision and Pattern Recognition*, CVPR, 1991.

C. HARRIS : Tracking with rigid objects. *In MIT Press*, 1992.

C. HARRIS et M. STEPHENS : A combined corner and edge detector. *In Alvey Vision Conference*, AVC, 1988.

R.I. HARTLEY : Estimation of relative camera positions for uncalibrated cameras. *In European Conference on Computer Vision*, ECCV, 1992.

R.I. HARTLEY : In defence of the 8-point algorithm. *In International Conference on Computer Vision*, ICCV, 1995.

R.I. HARTLEY et P.F. STURM : Triangulation. *Computer vision and image understanding*, 68(2):146–157, 1997.

R.I. HARTLEY et A. ZISSERMAN : *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

R.I. HARTLEY et A. ZISSERMAN : *Multiple View Geometry in Computer Vision*. Cambridge University Press, second édition, 2004.

J. HEIKKILA et O. SILVEN : A four-step camera calibration procedure with implicit image correction. *In Computer Vision and Pattern Recognition*, CVPR, 1997.

S.J. HENDERSON et S. FEINER : Evaluating the benefits of augmented reality for task localization in maintenance of an armored personnel carrier turret. *In International Symposium on Mixed and Augmented Reality*, ISMAR, 2009.

K. HIROSE et H. SAITO : Fast line description for line-based slam. *In British Machine Vision Conference*, BMVC, 2012.

K.L. HO et P. NEWMAN : Detecting loop closure with scene sequences. *International Journal of Computer Vision*, 74(3):261–286, 2007.

B.K.P. HORN et B.G. SCHUNCK : Determining optical flow. *Artificial intelligence*, 17(1):185–203, 1981.

P.J. HUBER : *Robust Statistics*. Wiley Series in Probability and Statistics. John Wiley & Sons, 1981.

D. KALKOFEN, E. MENDEZ et D. SCHMALSTIEG : Interactive focus and context visualization for augmented reality. *In International Symposium on Mixed and Augmented Reality*, ISMAR, 2007.

K. KANATANI, Y. SUGAYA et H. NIITSUMA : Triangulation from two views revisited : Hartley-sturm vs. optimal correction. *In British Machine Vision Conference*, BMVC, 2008.

T. KEMPTER, A. WENDEL et H. BISCHOF. : Online model-based multi-scale pose estimation. *In Computer Vision Winter Workshop*, CVWW, 2012.

G. KLEIN et D. MURRAY : Parallel tracking and mapping for small AR workspaces. *In International Symposium on Mixed and Augmented Reality*, ISMAR, 2007.

G. KLEIN et D. MURRAY : Improving the agility of keyframe-based slam. *In European Conference on Computer Vision*, ECCV, 2008.

K. KONOLIGE, M. AGRAWAL et J. SOLÁ : Large scale visual odometry for rough terrain. *In International Symposium on Robotics Research*, ISRR, 2007.

E. KRUPPA : Zur ermittlung eines objektes aus zwei perspektiven mit innerer orientierung. *SitzBer Akad Wiss Wien Math Naturw Abt IIa*, 122(122):1939–1948, 1913.

R. KUMAR et A.R. HANSON : Robust methods for estimating pose and a sensitivity analysis. *Computer Vision and Image Understanding*, 60(3):313–342, 1994.

V. KYRKI et D. KRAGIC : Tracking rigid objects using integration of model-based and model-free cues. *Machine Vision Application*, 22(2):323–335, 2011.

A. LADIKOS, S. BENHIMANE et N. NAVAB : A real-time tracking system combining template-based and feature-based approaches. *In International Conference on Computer Vision Theory and Applications*, VISAPP, 2007.

D. LARNAOUT, S. BOURGEOIS, V. GAY-BELLILE et M. DHOME : Towards bundle adjustment with gis constraints for online geo-localization of a vehicle in urban center. *In Joint 3DIM/3DPVT Conference*, 3DIMPVT, 2012.

V. LEPETIT et P. FUA : Monocular model-based 3d tracking of rigid objects : A survey. *Foundations and Trends in Computer Graphics and Vision*, 1(1):1–89, 2005.

V. LEPETIT et P. FUA : Keypoint recognition using randomized trees. *Transactions on Pattern Analysis and Machine Intelligence*, 28(9):1465–1479, 2006.

V. LEPETIT, F. MORENO-NOGUER et P. FUA : EPnP : An accurate solution to the PnP problem. *International Journal of Computer Vision*, 81(2):155–166, 2009.

S. LEUTENEGGER, M. CHLI et R. SIEGWART : Brisk : Binary robust invariant scalable keypoints. *In International Conference on Computer Vision*, ICCV, 2011.

K. LEVENBERG : A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics 2*, pages 164–168, 1944.

M. LHUILLIER : Incremental fusion of structure-from-motion and gps using constrained bundle adjustments. *Transactions on Pattern Analysis and Machine Intelligence*, 34(12):2489–2495, 2012.

P. LINDSTROM : Triangulation made easy. *In Computer Vision and Pattern Recognition*, CVPR, 2010.

H.C. LONGUET-HIGGINS : A computer algorithm for reconstructing a scene from two projections. *Nature*, 293(5828):133–135, 1981.

H.C. LONGUET-HIGGINS : *Readings in computer vision : issues, problems, principles, and paradigms*, chapitre A computer algorithm for reconstructing a scene from two projections, pages 61–62. 1987.

P. LOTHE, S. BOURGEOIS, F. DEKEYSER, E. ROYER et M. DHOME : Towards geographical referencing of monocular slam reconstruction using 3d city models : Application to real-time accurate vision-based localization. *In Computer Vision and Pattern Recognition*, CVPR, 2009.

P. LOTHE, S. BOURGEOIS, E. ROYER, M. DHOME et S. NAUDET-COLLETTE : Real-time vehicle global localisation with a single camera in dense urban areas : Exploitation of coarse 3d city models. *In Computer Vision and Pattern Recognition*, CVPR, 2010.

M.I.A. LOURAKIS et A.A ARGYROS : Is levenberg-marquardt the most efficient optimization algorithm for implementing bundle adjustment ? *In International Conference on Computer Vision*, ICCV, 2005.

M.I.A. LOURAKIS et A.A. ARGYROS : SBA : A Software Package for Generic Sparse Bundle Adjustment. *ACM Transactions on Mathematical Software*, 36 (1):1–30, 2009.

D.G. LOWE : Three-dimensional object recognition from single two-dimensional images. *Journal Artificial Intelligence*, 31(3):355–395, 1987.

D.G. LOWE : Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

C.P. LU, G.D. HAGER et E. MJOLSNESS : Fast and globally convergent pose estimation from video images. *Transactions on Pattern Analysis and Machine Intelligence*, 22(6):610–622, 2000.

E. MALIS et E. MARCHAND : Experiments with robust estimation techniques in real-time robot vision. *In International Conference on Intelligent Robots and Systems*, IROS, 2006.

E. MARCHAND, P. BOUTHEMY, F. CHAUMETTE et V. MOREAU : Robust real-time visual tracking using a 2d-3d model-based approach. *In International Conference on Computer Vision*, ICCV, 1999.

E. MARCHAND et F. CHAUMETTE : Virtual visual servoing : a framework for real-time augmented reality. *Computer Graphics Forum*, 21(3):289–298, 2002.

D. MARQUARDT : An algorithm for least-squares estimation of non linear parameters. *Journal of the Society for Industrial and Applied Mathematics*, 11(1):431–444, 1963.

J. MICHOT, A. BARTOLI et F. GASPARD : Bi-objective bundle adjustment with application to multi-sensor slam. *In 3D Data Processing Visualization and Transmission*, 3DPVT, 2010.

E.M. MIKHAIL, J.S. BETHEL et J.C. MCGLONE : *Introduction to modern photogrammetry*, volume 31. John Wiley & Sons, 2001.

K. MIKOLAJCZYK et C. SCHMID : An affine invariant interest point detector. *In European Conference on Computer Vision*, ECCV, 2002.

J. MODERSITZKI : *Numerical Methods for Image Registration (Numerical Mathematics and Scientific Computation)*. Oxford university press, 2004.

M. MONTEMERLO, S. THRUN, D. KOLLER et B. WEGBREIT : Fastslam : A factored solution to the simultaneous localization and mapping problem. *In Conference on Artificial Intelligence*, AAAI, 2002.

E. MOURAGNON, M. LHUILLIER, M. DHOME, F. DEKEYSER et P. SAYD : Real time localization and 3d reconstruction. *In Conference on Computer Vision and Pattern Recognition*, CVPR, 2006.

R.A. NEWCOMBE, A.J. DAVISON, S. IZADI, P. KOHLI, O. HILLIGES, J. SHOTTON, D. MOLYNEAUX, S. HODGES, D. KIM et A. FITZGIBBON : Kinectfusion : Real-time dense surface mapping and tracking. *In IEEE International Symposium on Mixed and Augmented Reality*, ISMAR, 2011.

D. NISTÉR : Preemptive ransac for live structure and motion estimation. *In International Conference on Computer Vision*, ICCV, 2003.

D. NISTÉR : An efficient solution to the five-point relative pose problem. *Transactions on Pattern Analysis and Machine Intelligence*, 26(6):756–777, 2004.

D. NISTÉR, O. NARODITSKY et J. BERGEN : Visual odometry. *In Computer Vision and Pattern Recognition*, CVPR, 2004.

D. NISTÉR et H. STEWENIUS : Scalable recognition with a vocabulary tree. *In Computer Vision and Pattern Recognition*, CVPR, 2006.

D. OBERKAMPF, D.F. DEMENTHON et L.S. DAVIS : Iterative pose estimation using coplanar points. *In Computer Vision and Pattern Recognition,*, CVPR, 1993.

A. PETIT, E. MARCHAND et K. KANANI : Tracking complex targets for space rendezvous and debris removal applications. *In International Conference on Intelligent Robots and Systems*, IROS, 2012.

J. PILET, V. LEPETIT et P. FUA : Real-time nonrigid surface detection. *In Computer Vision and Pattern Recognition*, CVPR, 2005.

J. PLATONOV, H. HEIBEL, P. MEIER et B. GROLLMANN : A mobile markerless ar system for maintenance and repair. *In International Symposium on Mixed and Augmented Reality*, ISMAR, 2006.

J. PLATONOV et M. LANGER : Automatic contour model creation out of polygonal cad models for markerless augmented reality. *In International Symposium on Mixed and Augmented Reality*, ISMAR, 2007.

M. PRESSIGOUT et E. MARCHAND : Hybrid tracking algorithms for planar and non-planar structures subject to illumination changes. *In International Symposium on Mixed and Augmented Reality*, ISMAR, 2006.

V.A. PRISACARIU et I.D. REID : Pwp3d : Real-time segmentation and tracking of 3d objects. *In British Machine Vision Conference*, BMVC, 2011.

L. QUAN et Z. LAN : Linear n-point camera pose determination. *Transactions on Pattern Analysis and Machine Intelligence*, 21(8):774–780, 1999.

R. RAGURAM, J.-M. FRAHM et M. POLLEFEYS : A comparative analysis of ransac techniques leading to adaptive real-time random sample consensus. *In European Conference on Computer Vision*, ECCV, 2008.

D. RAO : *CurveSLAM : utilizing higher level structure in stereo vision-based navigation*. Thèse de doctorat, University of Illinois at Urbana-Champaign, 2012.

F. ROTHGANGER, S. LAZEBNIK, C. SCHMID et J. PONCE : 3d object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints. *International Journal of Computer Vision*, 66(3):231–259, 2006.

P.J. ROUSSEEUW et A.M. LEROY : *Robust regression and outlier detection*. John Wiley & Sons, 1987.

E. ROYER : *Reconstruction 3D et localisation simultanée de caméras mobiles : une approche temps-réel par ajustement de faisceaux local*. Thèse de doctorat, Université Blaise Pascal-Clermont-Ferrand II, 2006.

E. RUBLEE, V. RABAUD, K. KONOLIGE et G. BRADSKI : Orb : An efficient alternative to sift or surf. *In International Conference on Computer Vision*, ICCV, 2011.

G.A.F. SEBER et C.J. WILD : *Nonlinear Regression*. Wiley Series in Probability and Statistics. John Wiley & Sons, 1989.

F. SERVANT, E. MARCHAND, P. HOULIER et I. MARCHAL : Visual planes-based simultaneous localization and model refinement for augmented reality. *In International Conference on Pattern Recognition*, ICPR, 2008.

G. SIMON : Tracking-by-synthesis using point features and pyramidal blurring. *In International Symposium on Mixed and Augmented Reality*, ISMAR, 2011.

G. SIMON et M.-O. BERGER : Pose estimation for planar structures. *Computer Graphics and Applications*, 22(6):46–53, 2002.

G. SIMON, V. LEPETIT et M.-O. BERGER : Computer vision methods for registration : Mixing 3d knowledge and 2d correspondences for accurate image composition. *In International Workshop on Augmented Reality*, IWAR, 1998.

S.N. SINHA, J.M. FRAHM, M. POLLEFEYS et Y. GENC : Gpu-based video feature tracking and matching. *In EDGE, Workshop on Edge Computing Using New Commodity Architectures*, 2006.

I. SKRYPNYK et D. G. LOWE : Scene modelling, recognition and tracking with invariant image features. *In International Symposium on Mixed and Augmented Reality*, ISMAR, 2004.

P. SMITH, I. REID et A. DAVISON : Real-time monocular SLAM with straight lines. *In British Machine Vision Conference*, BMVC, 2006.

N. SNAVELY, S. M. SEITZ et R. SZELISKI : Photo tourism : Exploring photo collections in 3d. *In Special Interest Group on GRAPHics and Interactive Techniques*, SIGGRAPH, 2006.

N. SNAVELY, S.M. SEITZ et R. SZELISKI : Modeling the world from internet photo collections. *International Journal of Computer Vision*, 80(2):189–210, 2008.

G. SOURIMANT, L. MORIN et K. BOUATOUCH : Gps, gis and video fusion for urban modeling. *In Computer Graphics International Conference*, CGI, 2007.

Charles V. STEWART : Robust parameter estimation in computer vision. *SIAM Reviews*, 41(3):513–537, 1999.

H. STRASDAT, J.M.M. MONTIEL et A.J. DAVISON : Real-time monocular slam : Why filter ? *In International Conference on Robotics and Automation*, ICRA, 2010.

P. F. STURM et S. J. MAYBANK : On plane-based camera calibration : A general algorithm, singularities, applications. *In Computer Vision and Pattern Recognition*, CVPR, 1999.

I.E. SUTHERLAND : Sketch pad a man-machine graphical communication system. *In Proceedings of the SHARE design automation workshop*, 1964.

R. SZELISKI et P.H.S. TORR : Geometrically constrained structure from motion : Points on planes. *In European Workshop on 3D Structure from Multiple Images of Large-Scale Environments*, 1998.

M. TAMAAZOUSTI, V. GAY-BELLILE, S. NAUDET-COLLETTE, S. BOURGEOIS et M. DHOME : Method for locating a camera and for 3d reconstruction in a partially known environment, 2011a.

M. TAMAAZOUSTI, V. GAY-BELLILE, S. NAUDET-COLLETTE, S. BOURGEOIS et M. DHOME : Nonlinear refinement of structure from motion reconstruction by taking advantage of a partial knowledge of the environment. *In Computer Vision and Pattern Recognition*, CVPR, 2011b.

M. TAMAAZOUSTI, V. GAY-BELLILE, S. NAUDET-COLLETTE, S. BOURGEOIS et M. DHOME : Real-time accurate localization in a partially known environment : Application to augmented reality on 3d objects. *In International workshop on AR/MR registration, tracking and benchmarking, ISMAR Workshop*, TrakMark, 2011c.

M. TAMAAZOUSTI, V. GAY-BELLILE, S. NAUDET-COLLETTE et M. DHOME : Raffinement non linèaire d'une reconstruction de type sfm dans un environnement partiellement connu. *In Congrés des jeunes chercheurs en vision par ordinateur*, ORASIS, 2011d.

M. TAMAAZOUSTI, V. GAY-BELLILE, S. NAUDET-COLLETTE et M. DHOME : Localisation prècise et temps rèel dans un environnement partiellement connu : application au suivi d'objet 3d peu texturè. *In Confèrence Reconnaissance des Formes et Intelligence Artificielle*, RFIA, 2012.

J.-P. TARDIF, Y. PAVLIDIS et K. DANIILIDIS : Monocular visual odometry in urban environments using an omnidirectional camera. *In International Conference on Intelligent Robots and Systems*, IROS, 2008.

J.D. TARDOS, J. NEIRA, P.M. NEWMAN et J.J. LEONARD : Robust mapping and localization in indoor environments using sonar data. *The International Journal of Robotics Research*, 21(4):311–330, 2002.

S. TAYLOR, E. ROSTEN et T. DRUMMOND : Robust feature matching in 2.3μs. *In IEEE CVPR Workshop on Feature Detectors and Descriptors : The State Of The Art and Beyond*, 2009.

C. TEULIERE, E. MARCHAND et L. ECK : Using multiple hypothesis in model-based tracking. *In International Conference on Robotics and Automation*, ICRA, 2010.

S. THRUN, W. BURGARD et D. FOX : *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

C. TOMASI et T. KANADE : Detection and tracking of point features. Rapport technique, Carnegie Mellon University Technical Report CMU-CS-91-132, 1991.

B. TRIGGS, P. F. MCLAUCHLAN, R. I. HARTLEY et A. W. FITZGIBBON : Bundle adjustment - a modern synthesis. *In International Workshop on Vision Algorithms Theory and Practice*, ICCVW, 2000.

T. TUYTELAARS et K. MIKOLAJCZYK : Local invariant feature detectors : a survey. *Foundation and Trends in Computer Graphics and Vision*, 3(3):177–280, 2008.

S. ULLMAN : The interpretation of visual motion. *In The MIT Press, Cambridge, MA*, 1979.

L. VACCHETTI et V. LEPETIT : Stable real-time 3d tracking using online and offline information. *Transactions on Pattern Analysis and Machine Intelligence*, 26(20): 1385–1392, 2004.

L. VACCHETTI, V. LEPETIT et P. FUA : Combining edge and texture information for real-time accurate 3d camera tracking. *In International Symposium on Mixed and Augmented Reality*, ISMAR, 2004.

R.G. VON GIOI, J. JAKUBOWICZ, J.M. MOREL et G. RANDALL : Lsd : A fast line segment detector with a false detection control. *Transactions on Pattern Analysis and Machine Intelligence*, 2010.

H.H. VU, R. KERIVEN, P. LABATUT et J.P. PONS : Towards high-resolution large-scale multi-view stereo. *In Computer Vision and Pattern Recognition*, CVPR, 2009.

D. WAGNER, G. REITMAYR, A. MULLONI, T. DRUMMOND et D. SCHMALSTIEG : Real-time detection and tracking for augmented reality on mobile phones. *Transactions on Visualization and Computer Graphics*, 16(3):355–368, 2010.

H. WANG et D. SUTER : Robust adaptive-scale parametric model estimation for computer vision. *Transactions on Pattern Analysis and Machine Intelligence*, 26 (11):1459–1474, 2004.

L. WANG, S. YOU et U. NEUMANN : Supporting range and segment-based hysteresis thresholding in edge detection. *In International Conference on Image Processing*, ICIP, 2008.

Z. WANG, F. WU et Z. HU : Msld : A robust descriptor for line matching. *Pattern Recognition*, 39:889–896, 2009.

T. WHELAN, M. KAESS, M.F. FALLON, H. JOHANNSSON, J.J. LEONARD et J.B. MCDONALD : Kintinuous : Spatially extended KinectFusion. *In RSS Workshop on RGB-D : Advanced Reasoning with Depth Cameras*, 2012.

H. WUEST, D. STRICKER et J. HERDER : Tracking of industrial objects by using cad models. *Journal of Virtual Reality and Broadcasting*, 4(1), 2007a.

H. WUEST, F. VIAL et D. STRICKER : Adaptive line tracking with multiple hypotheses for augmented reality. *In International Symposium on Mixed and Augmented Reality*, ISMAR, 2005.

H. WUEST, F. WIENTAPPER et D. STRICKER : Adaptable model-based tracking using analysis-by-synthesis techniques. *In Computer Analysis of Images and Patterns*, CAIP, 2007b.

G. XU, J. TERAI et H.Y. SHUM : A linear algorithm for camera self-calibration, motion and structure recovery for multi-planar scenes from two perspective images. *In Computer Vision and Pattern Recognition*, CVPR, 2000.

L. ZHANG et R. KOCH : Hand-held monocular slam based on line segments. *In Irish Machine Vision and Image Processing*, IMVIP, 2011.

Z. ZHANG : Determining the epipolar geometry and its uncertainty : A review. *International journal of computer vision*, 27(2):161–195, 1998.

Z. ZHANG : A flexible new technique for camera calibration. *Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

Table des matières

Table des figures

Liste des tableaux